ÓBUDAI EGYETEM
ÓBUDA UNIVERSITY

NEUMANN JÁNOS
INFORMATIKAI KAR

SZAKDOLGOZAT

OE-NIK	Hallgató neve:	**Sipos Levente Szabolcs**
2023	Hallgató törzskönyvi száma:	**T/006927/FI12904/N**

Óbudai Egyetem
Neumann János Informatikai Kar
Kiberfizikai Rendszerek Intézet

SZAKDOLGOZAT FELADATLAP

Hallgató neve:	**Sipos Levente Szabolcs**
Törzskönyvi száma:	T/006927/FI12904/N
Neptun kódja:	[redacted]

A dolgozat címe:

Decentralizált kommunikációs hálózat fejlesztése
Development of a decentralised communication network

Intézményi konzulens:	Lovas István
Külső konzulens:	
Beadási határidő:	2022. december 15.
A záróvizsga tárgyai:	Számítógép architektúrák IoT, beágyazott rendszerek és robotika specializáció

A feladat

Tervezzen és valósítson meg egy decentralizált kommunikációs hálózatot. A cél egy olyan hálózat kialakítása, amely könnyen bővíthető, és magas hibatűréssel rendelkezik. A hálózatban az adatok irányítását, és eszközök csatlakozását csomópontok végezzék. A csomópontoknak úgy kell összedolgozniuk, hogy egy egység kiesése ne jelentse a meghibásodott berendezéshez csatlakozó további hálózatok leszakadását. Tervezése során vizsgálja meg a biztonsági szempontokat. Biztosítani kell a felhasználók számára a megfelelő adatátviteli sebességet. Szükséges, hogy a felhasználóknak lehetősége legyen külső hálózatokkal való kommunikációra is.

A dolgozatnak tartalmaznia kell:

- a megoldandó feladat pontos leírását és részletes elemzését,
- a meglévő rendszerek vizsgálatának elemzését,
- biztonsági szempontok elemzését,
- az alkalmazni kívánt technológiák elemzését,
- a rendszertervet, döntések indoklásait,
- a megvalósítás leírását,
- a tesztelési tervet és a tesztek eredményeit.

Ph.

ÓBUDAI EGYETEM * Neumann János Informatikai Kar * Budapest

.....
Dr. Fleiner Rita
intézetigazgató

A szakdolgozat elévülésének határideje: **2024. december 15.**
(OE TVSz 55.§ szerint)

A dolgozatot beadásra alkalmasnak tartom:

.....
külső konzulens

.....
intézményi konzulens

KONZULTÁCIÓS NAPLÓ

Hallgató neve: Neptun kód: Tagozat:

Sipos Levente Szabolcs ... [redacted] ..... Nappali .....

Telefon: Levelezési cím (pl: lakcím):

[redacted]

Szakdolgozat / Diplomamunka címe magyarul:

Decentralizált kommunikációs hálózat fejlesztése .....

Szakdolgozat / Diplomamunka címe angolul:

Development of a decentralised communication network .....

Intézményi konzulens: Külső konzulens:

Lovas István..... .....

Kérjük, hogy az adatokat nyomtatott nagybetűkkel írja!

Alk.	Dátum	Tartalom	Aláírás
1.	2022.02.16	Témaválasztási konzultáció	
2.	2022.03.09	Szakirodalom áttekintése	
3.	2022.04.06	Lehetséges megvalósítás áttekintése	
4.	2022.05.04	Tartalmi, formai ellenőrzés	

A Konzultációs naplót összesen 4 alkalommal, az egyes konzultációk alkalmával kell láttamoztatni bármelyik konzulenssel.

A hallgató a Szakdolgozat I. / Szakdolgozat II. (BSc) vagy Diplomamunka 1 / Diplomamunka 2 / Diplomamunka 3 / Diplomamunka 4 tantárgy követelményét teljesítette, beszámolóra / védésre bocsátható.

Budapest, 2022. 05. 11.

.....
Intézményi konzulens

KONZULTÁCIÓS NAPLÓ

Hallgató neve: Neptun kód: Tagozat:

Sipos Levente Szabolcs.... [redacted] ..... Nappali.....

Telefon: Levelezési cím (pl: lakcím):

[redacted]

Szakdolgozat / Diplomamunka címe magyarul:

Decentralizált kommunikációs hálózat fejlesztése.....

Szakdolgozat / Diplomamunka címe angolul:

Development of a decentralised communication network .....

Intézményi konzulens: Külső konzulens:

Lovas István ....

Kérjük, hogy az adatokat nyomtatott nagybetűkkel írja!

Alk.	Dátum	Tartalom	Aláírás
1.	2022.09.14.	Specifikáció áttekintése	
2.	2022.10.12.	Rendszerterv áttekintése	
3.	2022.11.09.	Fejlesztés, tesztelés ellenőrzés	
4.	2022.12.05.	Tartalmi, formai ellenőrzés	

A Konzultációs naplót összesen 4 alkalommal, az egyes konzultációk alkalmával kell láttamoztatni bármelyik konzulenssel.

A hallgató a Szakdolgozat I. / Szakdolgozat II. (BSc) vagy Diplomamunka 1 / Diplomamunka 2 / Diplomamunka 3 / Diplomamunka 4 tantárgy követelményét teljesítette, beszámolóra / védésre bocsátható.

Budapest, 2022.12.07.

.....
Intézményi konzulens

HALLGATÓI NYILATKOZAT

Alulírott hallgató kijelentem, hogy a szakdolgozat saját munkám eredménye, a felhasznált szakirodalmat és eszközöket azonosíthatóan közöltem. Az elkészült szakdolgozatomban található eredményeket az egyetem és a feladatot kiíró intézmény saját céljára térítés nélkül felhasználhatja.

Budapest, 2022. 12. 14.

Sipos Levente Szabolcs

hallgató aláírása

Tartalom

1. Bevezetés

A szakdolgozatom célja egy decentralizált kommunikációs hálózat fejlesztése és megvalósítása. Ezt elsősorban meglévő technológiák és eszközök használatával tervezem kialakítani. A rendszer alapjaként csomópontok szolgálnak, amelyek képesek egymáshoz csatlakozni. A hálózat ezek között az eszközök között alakul ki. Csomópontokhoz kapcsolódhatnak felhasználói eszközök is, amelyek így képesek a kiépített hálózat részévé válni. A hálózat elsősorban vezeték nélküli technológiát használ a csatlakozások kivitelezéséhez.

A hálózat decentralizált tulajdonsága abban rejlik, hogy egyenrangú csomópontok kapcsolata alakítja ki, és nincsenek fő irányító egységek. A kialakítás előnye konvencionális megoldásokhoz képest a könnyű bővíthetőség és magas hibatűrő képesség. A csomópontok meghibásodása nem jelent problémát a hálózat számára, mivel más egységek könnyedén át tudják venni a kiesett berendezés helyét. Hiba esetén a hálózat nem válik használhatatlanná, mert a többi berendezés ugyanúgy folytatja tovább a működését. Csomópontok sűrű elhelyezése esetén fel se tűnhet a felhasználó számára az egyes egységek meghibásodása.

A decentralizált tulajdonság lehetővé teszi több különböző felépítésű eszköznek is, hogy csomópontként részt vegyen a hálózatban. Így a hálózathoz specializált egységek is könnyen csatlakozhatnak. A specializált egységek olyan funkciókat látnak el, amelyeket az alap eszközök nem képesek. Ilyen funkciók lehetnek például az internetkapcsolat biztosítása a hálózat számára, nagy távolságok áthidalására képes vezeték nélküli adóvevő létesítése, illetve a vezetékes csatlakozás lehetővé tétele. Ezek a specializált egységek megkönnyíthetnek bizonyos működési funkciókat, illetve felruházhatják a hálózatot a céloknak megfelelő tulajdonságokkal.

A csomópontok kialakításánál a célom a minél kisebb méret és minél alacsonyabb energiafogyasztás elérése, így elősegítve a sokrétű felhasználást. A hálózatok könnyű kiépítése révén alkalmas otthonokban és munkahelyeken is egyaránt. Szintén előnyös lehet használatuk olyan területeken, amelyeken konvencionális hálózatok kiépítése nem egyszerű. Mobilis hálózatok is kialakíthatók ezekből, méretüknek és alacsony fogyasztásuknak köszönhetően. Illetve olyan helyeken is alkalmazhatóak, amelyek nem rendelkeznek előre kiépített központi infrastruktúrával, mint elektromosság vagy internet.

2. Irodalomkutatás

2.1 Vezeték nélküli kommunikáció

Vezeték nélküli kommunikáció alatt olyan információközlést értünk, amely közvetlen fizikai kapcsolat nélkül történik két vagy több fél között.

Mivel a csomópontok és a felhasználói eszközök így kommunikálnak, illetve csatlakoznak egymáshoz, szükséges ismerni a technológia működését, tulajdonságait és határait ahhoz, hogy megfelelően integráljuk a rendszerbe.

Nagy előnye a fizikai kommunikációval szemben, hogy nagyobb távolságokat képes áthidalni technológiától függően közvetlen rálátás és fizikai kapcsolat nélkül.

2.1.1 Működés alapja

A vezeték nélküli hálózatok rádióhullámok segítségével képesek adatokat küldeni és fogadni. Ezt az adatmozgatást a jelenleg használt legtöbb eszköz 2,4 GHz-en és 5 GHz-en végzi. Modern eszközök és újabb szabványok segítségével ez a frekvencia már a 6 GHz is lehet. Vezeték nélküli kommunikációra alkalmas eszközök képesek ezeket a jeleket adatokká és az adatokat jelekké alakítani. A hálózat átviteli sebességét a frekvenciája határozza meg. Ez minél nagyobb, annál több adatot képes egy adóvevő küldeni, illetve fogadni.

Különböző szabványok léteznek, amelyek a technológia előrehaladtával, illetve a specifikus felhasználási módok szüksége miatt jöttek létre. Az eredeti szabvány száma IEEE 802.11. Későbbi változatai a szabvány után elhelyezett betűjelekkel megkülönböztethetőek. [16]

A vezeték nélküli hálózatokban is fennáll a veszélye a csomagütközéseknek. A vezetékes hálózatokhoz hasonlóan itt is szükséges az ütközések elkerülése, erre a különböző szabványok által implementált „ütközéselkerülési” (CSMA/CA) algoritmus felelős. Ez az algoritmus szabályozza, hogy minden küldő kizárólag akkor tudjon adatot küldeni, amikor a kommunikációhoz használt csatorna szabad. Amikor ez bekövetkezik, a küldő fél kibocsát egy adatcsomagot, és mindaddig vár, amíg a célállomás ezt le nem igazolja. Figyelembe kell venni a hálózat terhelését, minél több berendezés csatlakozik egy hálózathoz, annál több ilyen ütközés következik be, és annál jobban csökken a teljesítménye. [7]

2.1.2 Megvalósításhoz alkalmas szabványok

A különböző szabványok befolyásolják az egyes kapcsolatok tulajdonságát és teljesítményét, ezért fontos, hogy a megfelelő szabvány kerüljön implementálásra a végleges csomópontokban.

Szabvány	Maximális bitráta [Mbps]	Maximális távolság [m]	Frekvenciatartomány(ok) [GHz]
802.11b	11	35 – 140	2,4
802.11g	54	45 – 90	2,4
802.11n	600	70 – 250	2,4 – 5
802.11ac	1000-7000	70 – 250	5

1. Táblázat: Leggyakoribb 802.11 szabványok, és azok tulajdonságai. [2]

A felhasznált frekvenciatartomány sokban befolyásolja a rendszer viselkedését. Minél nagyobb ez az érték, annál nagyobb az adatátviteli képessége az adott hálózatnak. A magas frekvenciatartomány hátránya viszont, hogy nehezebben hatol át tárgyakon, falakon és egyéb objektumokon. Ha a rendszer olyan helyen kerül felhasználásra, ahol sok fal található, akkor célszerű az alacsonyabb, 2,4 Ghz-es frekvenciát használni. Ez az átviteli sebesség csökkenésével jár, viszont a jel stabilabb és erősebb lesz.

Csomópontok számára a 802.11n a legalkalmasabb szabvány. A szabvány képes mind a 2,4 GHz-es és az 5 GHz-es tartományban sugározni. Ez lehetővé teszi a sokrétű felhasználást. Ha nagyobb átviteli sebességre van szükség, akkor lehetőség van magasabb frekvenciatartományt alkalmazni, viszont, ha stabilabb és megbízhatóbb hálózat kell, akkor 2,4 GHz-es tartomány is elérhető. Olyan esetekben, amikor ez a szabvány nem áll rendelkezésre, megfelelő lehet a 802.11g szabvány is.

2.1.3 802.11s

A 802.11s vezeték nélküli szabvány specifikusan „mesh” hálózatok kialakítására lett létrehozva. A protokoll az OSI modell második rétegében működik. A létrehozott hálózaton minden csomópont úgy látja egymást, mintha egy virtuális „switch”-re lennének kapcsolódva. A protokoll támogat bármely harmadik rétegen működő technológiát az adatok közlésére, például IPv4 és IPv6. [3]

A szabvány 2011-ben jött létre, viszont 2012 óta a standard IEEE 802.11 szabvány része.

2.1.4 Wi-Fi Roaming

A technológia az adótornyokhoz hasonlóan lehetővé teszi, hogy a felhasználó számára a kiépített hálózat egy homogén egységként jelenjen meg. Segítségével az eszközök észrevétlenül átcsatlakozhatnak egyik csomópontról a másikra, így biztosítva a lehető legjobb jelerősséget.

A Wi-Fi Roaming nem a csomópontok feladata, hanem a hálózatra csatlakozó felhasználói eszközöké. Ezen eszközök mindegyike maga dönti el, hogy mikor és hova szeretne átcsatlakozni. Mivel nincs befolyása a rendszernek ebbe a folyamatba, a hálózatot úgy kell kiépíteni, hogy az újracsatlakozások minél könnyebben történjenek. Ezt a csomópontok helyes elhelyezésével érhetjük el. Egységek akkor helyezkednek el

optimálisan, ha kettő hálózat mindössze 15-20%-ban fedi le egymást, illetve az átfedett területen a jelek erőssége 67 dBm-nél gyengébb. Ennek fontossága abban rejlik, hogy amikor az átfedés túl nagy, akkor a csatlakozott eszközök ugrálhatnak két hálózat között. Ha viszont túl kevés, akkor nem csatlakozik megfelelően a felhasználói eszköz a másik hálózathoz, így a felhasználó számára gyenge jelerősség lesz tapasztalható, esetlegesen teljesen le is kapcsolódik a hálózatról az eszköze. [1]

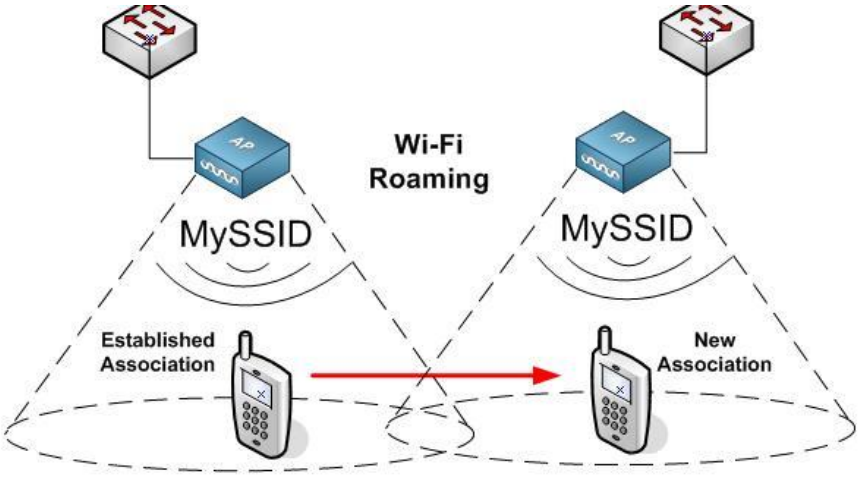
Az átcsatlakozáshoz szükséges, hogy mindegyik csomópont ugyanazzal az SSID-vel és hitelesítési móddal rendelkezzen. A felhasználói eszközök egy háromfázisos folyamat segítségével kapcsolódnak újra.

Fázisok:

1. Keresés,
2. Authentikáció,
3. Újracsatlakozás és újratársítás.

A folyamatok a következőket hajtják végre:

- Az első fázis gyenge jel esetén indul el, ennek során a felhasználói eszköz keresőcsomagokat küld. Ezek a csomagok alternatív access pointokat keresnek, ahova gyenge jel esetén a kliens átválthat.
- Második fázisban a kliens a kiválasztott access pointnak küld egy kérést, amelyben igényli, hogy az állomás hitelesítse. A kérés után várakozik annak a válaszára.
- Harmadik fázisban, amikor az access point elfogadja a kliens kérését, a felhasználói eszköz küld még egy kérést, amelyben az újracsatlakozást és az újratársítást kéri. Az újratársítás során az új access point, amelyhez már a kliens eszköz csatlakozik, küld egy csomagot a réginek, amellyel frissíti annak útválasztó tábláját. [1]



1. ábra: Wi-Fi Roaming újratársítás.

A decentralizált hálózatban ez a megvalósítás előnyös lehet, mivel a már kiépített hálózat így könnyen bővíthető, és a felhasználó számára is jobb használhatóságot biztosít, ugyanis nem kell minden egyes csomóponthoz manuálisan csatlakozni.

Esetleges hátrány lehet, ha a csomópontok nem megfelelően vannak elhelyezve, ami a már említett túl gyakori átcsatlakozást eredményezheti. Ügyelni kell viszont arra is, hogy a csomópontok ne legyenek túl távol, mivel ebben az esetben az átcsatlakozás nem következhet be megfelelően.

2.2 Hálózat topológiák

Hálózat topológiája alatt a hálózat felépítését értjük, tehát azt, hogy hogyan kapcsolódnak az egyes adóvevők egymáshoz.

Mivel decentralizált hálózatot valósít meg a rendszer, ezért minden olyan topológiát kizárhatunk a vizsgálatból, amely egy központi elemre épül, ilyen például a csillag felépítésű hálózat. Szintén figyelmen kívül lehet hagyni olyan struktúrákat, amelyek megkövetelik, hogy a csomópontok egymás után sorrendben küldjenek csomagokat.

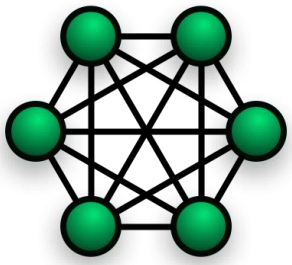
A fenti szempontok alapján a következő topológiák megfelelőek a kívánt működés eléréséhez:

- Teljes háló topológia,
- Részleges háló topológia,
- Fa topológia.

2.2.1 Teljes háló topológia

A teljes háló topológia minden csomópontot összeköt egymással. Ez a felépítés rendelkezik a legnagyobb hibatűréssel, mivel itt az egyes csomópontok kapcsolata nem függ másik egységtől. Illetve ez a hálófajta sok redundáns csatlakozással rendelkezik.

A kialakításából adódóan ez a legköltségesebb felépítés. A csatlakozások száma ennél a hálózatnál a legtöbb.



2. ábra: Teljes háló topológia.

Nem szükséges a csomagok irányítása, mert mindegyik csomópont közvetlenül elér bármelyik másik egységet.

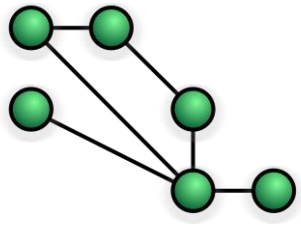
A kapcsolatok száma a hálózatban:

$$\frac{n(n-1)}{2} \tag{1}$$

Mivel a topológia fenntartásához mindegyik berendezésnek kapcsolatban kell állnia a másikkal, ezért csak kisebb helyiségekben lenne alkalmazható. Ha valamilyen okból egy csomópont nem tud kapcsolódni a többi egységhez, akkor felborul a hálózat kommunikációja. Emiatt ez a kialakítás nem alkalmas a felhasználásra. Ha a megvalósítás azt igényli, hogy a hálózatban ne legyenek hurkok, akkor szintén nem jó ez a kialakítás a végleges egység hálózatához.

2.2.2 Részleges háló topológia

A részleges háló topológia egy leegyszerűsített teljes háló topológia. Eltér az előbbi hálózatfajtától abban, hogy itt nincs mindegyik csomópont egymáshoz csatlakoztatva. Ez a kialakítás költséghatékonyabb a teljes változatnál, mivel a kapcsolatok száma is kisebb. A hálózatban szintén megjelenhetnek hurkok, amelyek nem feltétlenül előnyösek a végleges hálózat kialakításakor. Egységek meghibásodása esetén előfordulhat, hogy a hálózat több részre szakad. Ennek elkerülését az egységek sűrű installációjával lehet megoldani, mivel így alternatív útvonalak is létrejönnek. A megvalósítás a hurkokat leszámítva megfelelő lehet a hálózat kialakításához.

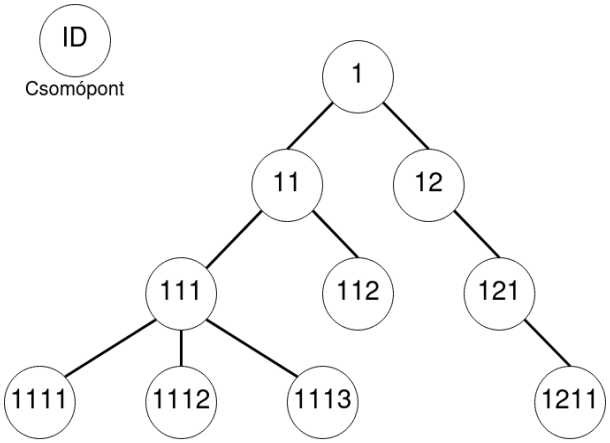


3. ábra: Részleges háló topológia.

2.2.3 Fa topológia

A fa topológia nem teljesen decentralizált hálózatfelépítés, mivel a csomópontok egymásra épülnek. Hibatűrése hasonló, mint a részleges háló topológiáé. Konvencionális fa felépítésű hálózatoknál, ha egy berendezés meghibásodik, akkor minden olyan egység kiesik, amely a meghibásodott csomópontra csatlakozott. Ha az egyes berendezések elég sűrűn vannak elhelyezve, akkor a csomópontok vezeték nélküli tulajdonsága miatt egy egyszerű átcsatlakozással kiküszöbölhető a teljes ág leválása. A topológia hasonlít a részleges háló felépítéshez, viszont ebben a hálózattípusban nem találhatóak hurkok. Egyes csomópontok beazonosítása egyszerű, mivel mindegyik egység egy bizonyos helyen szerepel a hálózatban.

Újracsatlakozás esetén a leszakadt ág legelső működő egysége újracsatlakozik a számára elérhető legerősebb jelű csomóponthoz, miközben figyeli, hogy ne csatlakozzon a saját ágán lévő berendezések hálózatára. Azonosítók miatt a csomagok irányítása is könnyebb lehet, mivel ez alapján az egyes egységek pontosan tudják, hogy hova kell tovább küldeniük az érkező adatokat. Ennek a hálózatfajtának a felépítése a legköltséghatékonyabb. Ez a topológia rendelkezik az egy csomópontra eső legkevesebb csatlakozással. A fa topológia is megfelelő lehet a hálózat kialakításához.



4. ábra: Fa topológia, és csomópontok beazonosíthatósága.

2.3 Painless Mesh

2.3.1 Mi is a Painless Mesh?

A Painless Mesh egy könyvtár, amely segítségével egyszerűen kialakíthatóak mesh hálózatok. Célja, hogy a felhasználó könnyedén tudjon hálózatokat kialakítani anélkül, hogy annak struktúrájával foglalkoznia kéne. A megvalósítás az esp8266 és az esp32 platformokat célozza meg. Nem igényel központi irányítót, routereket és esetleges előre tervezést. A különálló eszközök képesek maguktól, külső beavatkozás nélkül egy teljesen működőképes hálózatba rendeződni. A hálózathoz csatlakoztatható egységek és felhasználó eszközök száma csak az egyes csomópontok memóriájának méretétől függ. A projekt alapjaként a korábbi „easyMesh” (https://github.com/Coopdis/easyMesh) szolgált. [13]

2.3.2 Működési elve

A hálózat JSON objektumokat használ kommunikációra TCP/IP csomagok helyett. Ez az olvashatóság elősegítése és az adatok különböző alkalmazásokba való integrálása céljából lett így megvalósítva. Adatcsomagokat „broadcast” vagy „unicast” módon tudnak a különböző egységek küldeni egymásnak.

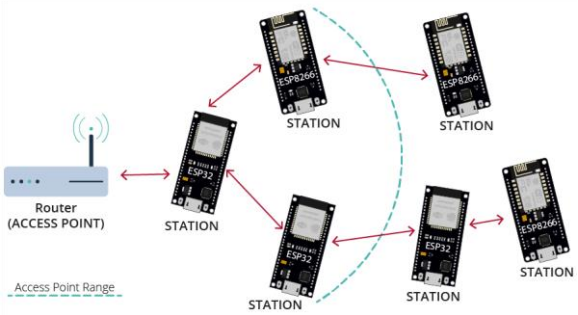
Mindegyik ESP eszköznek van egy egyedi hardver azonosítója, amely alapján az egyes csomópontok beazonosíthatóak.

A hálózati kommunikációban kétféle csomagfajta kerül felhasználásra. Ezek a csomagfajták az irányító és a felhasználói csomagok. Az irányító csomagok kezelik a csomópontok közötti idő szinkronizálását és a csomagok irányítását, a felhasználói csomagok pedig az adatok közlését. [11]

A JSON vezérlő csomagok tartalmazzák a célállomás és a küldő állomás azonosítóját, a csomag fajtáját, ezek mellett a csomag küldésének idejét. Ha a csomag egy másik egységen keresztül halad, akkor a keresztülhaladás ideje is hozzáadódik ehhez a csomaghoz. A felhasználói csomagok felépítése is hasonló. Szintén tartalmazzák a küldő és a célállomás azonosítóját, a csomag fajtáját. Abban tér el, hogy egy üzenet adattagot is tartalmaz, amellyel képes adatokat közölni a többi egység számára. [11]

2.3.3 Hálózat felépítése

Az ESP eszközök limitált erőforrása miatt a hálózatban a körök kialakulását aktívan kerüli a rendszer. Ez a megoldás a hálózat leegyszerűsítését is eredményezi. A kialakított hálózatot egy fa topológiához, vagy egy módosított csillag topológiához lehet hasonlítani. A painless mesh hálózatban mindegyik eszköz egyszerre tölti be az access point szerepét és mindegyik eszköz egy csomópont is, amelyhez további ESP eszközök csatlakozhatnak. Mindegyik csomópont ismeri a teljes hálózat topológiáját, ez a csatlakozott eszközökben 3 másodpercenként frissül. A topológiai frissítésben közli mindegyik berendezés a többi csomóponttal, hogy mely egységek csatlakoznak hozzá közvetlenül és közvetetten. Az a csomópont, amely még nem csatlakozik egyetlen másik csomóponthoz se, folyamatosan keres egy hálózatot, amihez csatlakozhat. Az egységek mindig a legerősebb jellel rendelkező olyan hálózathoz csatlakoznak, amely még az általuk felépített topológiában nem szerepel. Ez akadályozza meg, hogy körök alakuljanak ki a hálózatban, biztosítva azt, hogy mindig csak egy úton legyenek elérhetőek a hálózat egyes tagjai. [17]



5. ábra: Painless Mesh által kialakított hálózat.

2.3.4 Előnyök

Mivel az egyes csomópontok ESP eszközök, ezért egy hálózatot nagyon olcsó kiépíteni, illetve bővíteni. Energiafelhasználása alacsony. Egyes eszközök specifikusan csak a hálózat működtetésével foglalkoznak, tehát semmilyen háttérfolyamat nincs jelen.

Az egyes csomópontok mérete kicsi, tehát könnyedén beépíthető akár más rendszerekbe is. Szükség esetén egyszerű a szállítása is.

2.3.5 Hátrányok

Alacsony teljesítménnyel rendelkeznek. Az ESP eszközök kis teljesítményű processzorral vannak felszerelve, ez a csomagok irányítását is korlátozza. A legnagyobb hátrányt a kommunikáció fajtája jelenti, mivel a JSON alapú kommunikáció korlátozó faktor. A painless mesh rendszer egyszerűbb érzékelőknek megfelelő hálózatot épít ki, amely csak mért értékek közlésére elegendő, esetlegesen szöveges adatok küldésére. Nagyobb sávszélességet igénylő rendszerekhez, mint például videóközvetítésre nem elegendő a hálózat adatátvitele.

2.3.6 Kapcsolat a szakdolgozathoz

A projekt sokban hasonlít a célként kitűzött megvalósításhoz. Felépítésében hasonló rendszer kerül implementálásra a végleges rendszerben. Viszont a hálózat átviteli sebessége és a JSON alapú kommunikáció nem megfelelő. Internetelérést nem tud biztosítani a csomópontokhoz csatlakoztatott eszközök számára, csakis egyszerű szöveges adatközlésre alkalmas. A hálózat „öngyógyító” tulajdonsága és az újracsatlakozás menete hasznos lehet végleges kialakításban.

2.4 B.A.T.M.A.N. advanced

A B.A.T.M.A.N advanced (Better Approach to Mobile Ad-hoc Networking, batman-adv a továbbiakban) egy irányítási protokoll, ami nagyon hasonlít működésében és felépítésében a 802.11s protokollhoz. A Linux kernel tartalmazza 2011-óta, kernelmodulként működik, ezzel is gyorsítva a hálózat működését. Az OSI modell második rétegében történik a csomagok irányítása. A felhasználó számára ezt úgy jeleníti meg, mintha egy LAN-on lenne a hálózaton található összes eszköz. Célja mesh hálózatok könnyű kiépítésének elősegítése. [5]

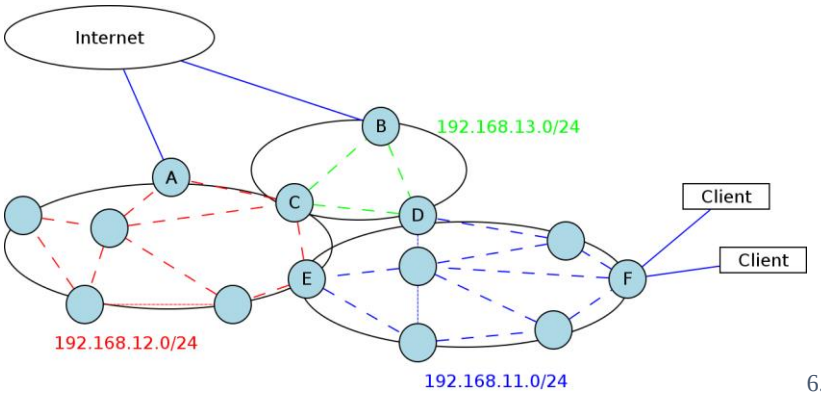
2.4.1 Működési elv

A protokoll az OSI modell második rétegében működik, tehát nem IP alapú hanem Ethernet kereteket használ a kommunikációra. Szimulál egy virtuális „switch”-et (kapcsolót), ami a hálózaton található csomagokat irányítja. Ennek a megvalósításnak köszönhetően bármilyen változás a hálózatban (csomópontok lecsatlakozása, meghibásodása) nem fogja befolyásolni annak működését. A szimulált kapcsoló segítségével az összes eszköz úgy érzékeli, mintha egy hálózaton lennének. Nagy előnyt jelent, hogy akármilyen protokoll futtatható a batman-adv felett. Lehetséges felhasználható példa protokollok, amik hasznosak lehetnek a rendszer használata közben: IPv4, IPv6, DHCP. [5]

Egy batman-adv hálózatra képes access point az általa szolgáltatott hálózaton található eszközöknek is képes a kiépített mesh rendszerhez hozzáférést nyújtani. Ilyenkor a

hálózatot szolgáltató interfész „bridge” állapotba kerül a batman-adv által használt interfésszel.

A protokoll, mivel OSI második rétegbeli ezért nem kezel alhálózatokat. Ha szükség van alhálózatokra, akkor a mesh hálózatot több részre kell bontani. [4]



6. ábra: B.A.T.M.A.N. Advanced hálózat felépítése. és alhálózatokra bontás.

2.4.2 Előnyök

Könnyen konfigurálhatóak a hálózatok. Linux rendszerek natívan támogatják. Magas átviteli sebességet is képes biztosítani, mivel ez csak a csomópontok teljesítményétől függ. OpenWrt, illetve hasonló router firmware-ek is támogatják, tehát ha szükséges, egy már előre felépített hálózatot is ki lehet bővíteni. Lehetőséget nyújt a protokoll internetelérés biztosítására a mesh hálózat eszközei számára, illetve a protokollt nem implementáló rendszerek is be tudnak lépni a hálózatba „bridgek” segítségével.

A batman-adv által létrehozott hálózat öngyógyító, önformáló, és önrendező, tehát a kezdeti konfigurálás után az egyes egységek maguktól kezelik a csatlakozásokat. Lehetőség van Raspberry Pi, és egyéb SBC (Single-board computer) eszközökön is implementálni, amennyiben minimum kettő hálózati interfész áll rendelkezésre.

A hálózatok mérete csak a benne található egység erőforrásaitól, számítási képességeitől, és memóriáitól függ, ez egyben hátrány is lehet, mivel a kisebb fogyasztású rendszerek korlátozó elemek lehetnek.

Konfigurálása egyszerű parancssoros utasítások segítségével történik.

2.4.3 Hátrányok

Alhálózatok kialakítását a hálózat tényleges szétdarabolásával lehetséges megoldani. Egy csomópont kiesését lassan érzékelheti a rendszer, mivel először azt feltételezi a kimaradt csomópontról, hogy csak átmeneti hiba történt. [4]

A protokoll külön biztonsági megoldásokat nem tartalmaz, azokat külön implementálni kell más rétegeken. [4]

Újraindítások között a konfigurációja nem permanens. Emiatt be kell vezetni egy indítási szkriptet, ami minden elinduláskor lefut és beállítja a rendszert.

2.4.4 Kapcsolat a szakdolgozathoz

A céloknak megfelelő mesh hálózatot épít ki, emellett könnyen konfigurálható. Rengeteg platformon elérhető. Mivel a Linux kernel is támogatja, ezért a protokoll használata nem lassítja a csomagok közvetítését. Akármilyen eszközön működik, amelyen Linux rendszer futtatható, és rendelkezik hálózati interfészekkel.

2.5 Hostapd

A hostapd (host access point daemon) egy nyílt forráskódú program, amely Linux és BSD alapú operációs rendszereken működik. A program képes arra alkalmas hálózati interfészeket access pointokká alakítani.

Konfigurálása egy ezt a célt szolgáló fájlban lehetséges. A beállítások jól dokumentáltak, és a módosításuk is egyszerű. A módosított beállítások a program újraindításakor lépnek életbe. A hálózatok titkosítását is lehetőségünk van konfigurálni. [6]

A létesített hálózatot nagy mélységben lehet személyre szabni.

Linux systemd szolgáltatások segítségével könnyedén elindítható csomópontok indításakor.

2.6 Dnsmasq

Egy olyan nyílt forráskódú program, amelynek célja DNS és DHCP szolgáltatások létesítése. Megvalósításhoz nem feltétlenül szükséges, mivel lehetséges, hogy a decentralizált hálózat nem IP alapon fog működni. Ha a hálózat interneteléréssel rendelkezik akkor a Dnsmasq elengedhetetlen abban az esetben, ha a csatlakozást biztosító eszköz nem szolgál ilyen funkciókkal.

Konfigurációja erre szolgáló fájlok módosításával lehetséges. Képes a programot futtató eszközön található „hosts” fájl tartalma alapján a DNS kéréseket módosítani, amely előnyös lehet, ha egy specifikus lokálisan futtatott oldalt szeretnénk elérhetővé tenni az eszközre csatlakozó kliensek számára.

Használata esetén szükséges a többi eszköz összehangolása, mivel nem előnyös, ha több ilyen szolgáltatás működik egy hálózaton.

3. Konfigurációs felület

Ahhoz, hogy a felhasználó módosítani tudja a hálózat működését, szükséges, hogy valamilyen felületen konfigurálnia tudja azt. Célszerű ezt megvalósítani olyan módon, amely lehetővé teszi olyan személyek számára is ennek végrehajtását, akik mögöttes ismeretekkel nem rendelkeznek a rendszer működéséről. A legegyszerűbb külső hozzáférést webes felületen keresztül lehetséges biztosítani. Ez a megoldás lehetővé teszi egy hálózaton található eszköz számára a csomópont címének ismeretében az egyszerű grafikus konfigurációt.

Ellenben, ha a felhasználó mélyebb ismeretekkel rendelkezik, és személyre szeretné szabni a konfigurációk finomhangolásával a hálózat működését, akkor erre is lehetőséget kell nyújtani számára.

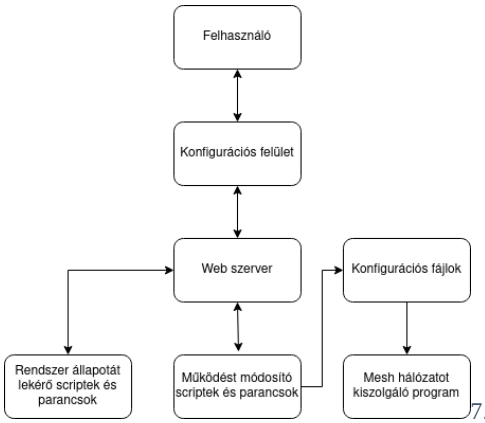
A felhasználónak lehetősége van a webes konfigurációs felület kikapcsolására, mivel ez nem mindenki számára szükséges, és felesleges erőforrásokat fogyaszthat a működtetése.

3.1 Webes felület

Webes felületen történő konfiguráció megvalósítása több program és szolgáltatás összehangolását igényli. A felület fenntartása a rendszer energiaigényét növelheti, és a teljesítményét minimális mértékben befolyásolhatja.

Szükséges a biztonsági szempontokra is ügyelni, mivel a webes felület a beállítások módosításához szkripteket, vagy parancsokat futtat. Ha ezen parancsokat, szkripteket illetéktelen személy módosítani képes, akkor sérülhet a hálózat biztonsága.

Mivel egyetlen program nem képes teljesen kiszolgálni a felület megjelenítését, működtetését és elérhetőségének biztosítását, ezért ezt a feladatot több kisebb részre bontottam. Ezek a részek a weboldal kiszolgálása, a weboldal megjelenítése, és a hálózat konfigurálása, esetlegesen egyéb funkciók biztosítása.



7. ábra: Webes felület rétegződése és felépítése.

3.1.1 Konfigurációs felület

A konfigurációs felületet egy weblap formájában tervezem megvalósítani. A weblap segítségével a felhasználó interakcióba képes lépni a rendszerrel, és ezáltal a csomópont működését tudja módosítani. Weboldallal történő interakció esetén nem elegendő az egyszerű statikus HTML oldalak használata.

Weboldalak interaktívvá tételéhez a legismertebb és legtöbbet használt nyelvek a JavaScript és a PHP.

Mivel a JavaScript magában csak kliens oldali működést képes biztosítani, ezért elengedhetetlen, ha egyedül ezt szeretnénk használni, hogy NodeJS-t alkalmazzunk, hogy biztosítani tudjuk a szerver oldali funkciókat is.

PHP esetén lehetséges kódokat a weboldalakba ágyazva alkalmazni, ezáltal a működést is biztonságosabbá lehet tenni. A kliens oldalon így nem lehet megnézni a lefordított és beágyazott PHP kódot. Nagy előnye, hogy képes a szerver oldalon is működni, tehát képes magában, más környezet használata nélkül a csomópont működését módosítani.

PHP és JavaScript nyelveket Jquery Ajax segítségével össze lehet kötni, és így is el lehet érni a kívánt működést. Ennek a megoldásnak előnye, hogy viszonylag egyszerűen és kevés további csomag telepítésével lehet a felhasználói felületet megoldani.

3.1.2 Webszerver

Webszerver biztosítja a konfigurációs weblapok elérhetőségét a csomópontokon, és továbbítja a rendszer felé a felhasználótól kapott kéréseket és utasításokat. Leggyakoribb webszerverek az Apache HTTP Server és az Nginx. A kettő megvalósítás közül bármely megfelel, mivel a weboldal nem rendelkezik majd nagy forgalommal, és mind a kettő konfigurálása egyszerű.

3.2 Mélyebb konfigurációt biztosító megoldás

A felhasználónak lehetősége van minden konfigurációs fájlt kézzel szerkeszteni, ez sokkal nagyobb fokú szabadságot biztosít abban az esetben, ha ért a rendszerhez. A konfiguráció egy egyszerű SSH (Secure Shell) kapcsolaton keresztül történik. A megvalósítás egyszerűbb felépítésű, energiatakarékosabb, és biztonságosabb is mint a webes felület használata, mivel ebben az esetben csupán egy SSH szervernek kell futnia a csomópontokon. Lehetősége van a weboldal konfigurációjához használható szkriptek futtatására is, amivel esetenként gyorsabb eredményt lehet elérni, mint a weboldalon történő konfigurációval.

4. Felhasználható hardverek

4.1 Szempontok

A célok megvalósítása érdekében a kisebb méret előnyt jelent, mivel megkönnyíti a hordozhatóságot.

Meglévő eszközök használata csökkenti a hálózatok kiépítésének és karbantartásának költségét, viszont így más gyártókra vagyunk utalva.

Előny, ha a hardver hálózati interfésszel/interfészekkel rendelkezik, viszont elfogadható az is, ha ez csak külső berendezések segítségével érhető el. Külső hálózati interfész olyan esetben is szükséges, ha a beépített antenna teljesítménye nem megfelelő. Ebben az esetben a beépített interfész akár egy biztonsági megoldást is képezhet, ha a fő egység meghibásodik.

A kis energiafogyasztás fontos szempont, mivel szükséges lehet az egyes csomópontok működtetése akár akkumulátorról is.

4.2 Átlagos router specifikációk

A megfelelő hardverkövetelmények kiválasztásához egy középkategóriás routerhez hasonlítottam a felhasználható eszközöket.

4.2.1 TP-Link Archer C7 AC1750 (v5)

Az Archer C7 egy középkategóriás router, amely támogatja a mesh hálózatok kiépítését dedikált TP-Link OneMesh eszközök segítségével. Rendelkezik Ethernet „switch”-csel, ami a hálózaton a csomagok irányítását segíti. Ez egy specifikus komponens, amely általában hálózati berendezésekben található meg. Hiánya csökkentheti a hálózat sebességét, viszont egy elég erős processzorral ezt a hátrányt minimalizálni lehet.

Az eszköz egy Qualcomm Atheros QCA9558 ARM processzort használ, amely 720 MHz-en működik. Tartalmaz 128 MB RAM-ot és 8 MB flash memóriát. Vezeték nélküli antennái képesek 2,5 és 5 GHz-es tartományban is működni. Szabványok terén képes a 802.11 „a”, „b”, „g”, „n” és „ac” verzióit kezelni. Maximális átviteli sebesség 2,5 GHz-es tartományban 450 Mbps, 5 GHz-en pedig 1300 Mbps. [14] Saját zárt forráskódú operációs rendszerrel rendelkezik, de lehetséges OpenWrt működtetése is az eszközön. [15]

8. ábra: TP-Link Archer C7.

4.2.2 Követelmények

Mivel a csomagok irányítására nem lesz külön Ethernet switch a választott hardverben, ezért fontos az processzor teljesítménye.

A belső memória mérete nem a legfőbb szempont, a legtöbb felhasználható hardver eszköz valószínűleg elegendővel rendelkezik. Minimum az 1 GB RAM, mivel így elegendő memóriája lesz a hálózati topológia és a bennük szereplő eszközök eltárolására, és esetlegesen más programok működtetéséhez is.

Vezeték nélküli teljesítménye minimálisan 50-100 Mbps kell legyen, ezt 2,5 GHz-en kell elérnie, viszont előnyt jelent, ha 5 GHz-en is képes adatokat közvetíteni.

4.3 Single-board számítógépek (SBC)

Ezek az eszközök egy teljes működő számítógépet valósítanak meg egyetlen nyáklapon. Általában kis formafaktorral rendelkeznek, olcsók, a fogyasztásuk is kellően alacsony. Teljesítményük gyengébb, mint a hagyományos számítógépeké, de hálózati funkciók kiszolgálására elegendőek lehetnek.

Jellemzőek az ARM típusú processzorok ezekben az eszközökben, ezért operációs rendszerek terén limitált választási lehetőséget nyújtanak. Léteznek viszont x86 és x64 architektúrával rendelkező processzoros változatok is, viszont ezek energiafogyasztása jelentősen magasabb.

4.3.1 Odroid XU4

Egy SBC, amely eredetileg egy fejlesztőpanelként szolgált, de a hobbi körökben is elterjedté vált.

9. ábra: Odroid XU4.

Az eszköz egy 2 Ghz-es Samsung Exynos5422 Cortex™-A15 és egy Cortex™-A7 8 magos ARM processzorral rendelkezik. Lehetséges Linux alapú operációs rendszert futtatni rajta. A támogatott Linux kernel verzió az 5.4 LTS. Teljesítménye megfelelő lenne az elvárt működéshez.

Belső RAM-ja egy 2 GB-os LPDDR3 egység.

Kettő USB 3.0 és egy USB 2.0 porttal rendelkezik. Az egység nincs külön felszerelve Wi-Fi modullal, ezért a külső antennák ezen portok segítségével csatlakoznának. Gigabit Ethernet csatlakozóval is rendelkezik, amely csomóponti működés során nem kerül felhasználásra, viszont specifikus működési esetekben használható lehet. [10]

Energiaigénye 20 W, amit egy „barrel” csatlakozóval lehet az egységnek táplálni. Ez 5 V-on és 4 A-en történik.

Ára $50-$60 körül mozog. Ez megfelelő ár hálózatok könnyű felépítéséhez és meghibásodott egységek cseréjéhez.

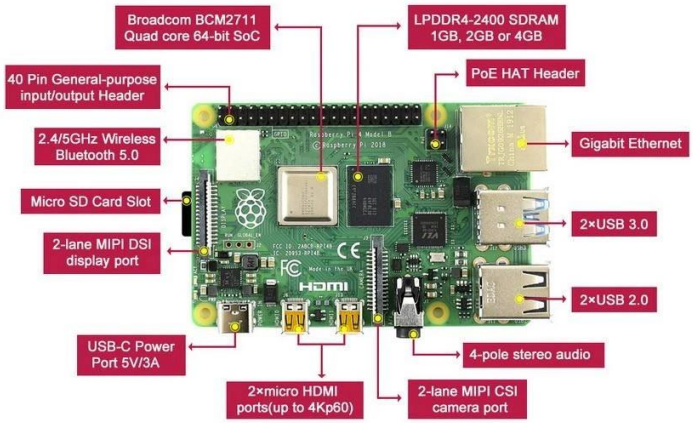
Kis formafaktorral rendelkezik, mérete körülbelül 83×58×20 mm. [10]

Előnyös ez a berendezés, mivel erőteljes processzorral rendelkezik a többi ilyen árkategóriában található SBC-hez képest.

Hátrányt jelent azonban a kisebb elterjedtsége. Kisebb támogatottsággal és felhasználói bázissal rendelkezik, mint például a Raspberry eszközök. Ez megnehezítheti a fejlesztést, mivel kisebb támogatottsággal rendelkeznek az egyes felhasználásra kerülő alkalmazások.

4.3.2 Raspberry Pi 4 Model B

Szintén egy SBC, amely főleg hobbiszintű felhasználásra lett kifejlesztve, viszont elég elterjedté vált. Kis mérete miatt alkalmas lenne a kitűzött feladatra. A 4-es modell B verziója a legújabb eszköz.



10. ábra: Raspberry Pi 4 Model B.

Egy Broadcom BCM271, és egy 64-bites ARM Quad core Cortex-A72 v8 processzorral rendelkezik, amely 1,5 GHz-en működik, de 2 GHz is elérhető belső konfigurálással.

Elsősorban Linux alapú operációs rendszerek preferáltak az eszközön. Egy saját Debian Linux alapú disztribúcióval is rendelkezik, ez a Raspberry Pi OS.

RAM-ja modellfüggő, létezik 1, 2, 4, illetve 8 GB-os verzió, ezek küzöl bármelyik megfelelő. A választási faktor az eszköz ára, mivel minél több RAM-mal van felszerelve, annál többe kerül.

Rendelkezik egy vezetékes Gigabit Ethernet csatlakozóval és egy vezeték nélküli 2,4 GHz, valamint 5 GHz-es Wi-Fi adóvevővel, ezek mellett Bluetooth 5.0 adóvevővel is fel van szerelve.

Kettő-kettő USB 3.0 és 2.0 porttal rendelkezik, ezek az erősebb Wi-Fi antenna, illetve egyéb perifériák csatlakoztatásához biztosítanak lehetőséget.

Egy 40 pinnel rendelkező GPIO fejjel rendelkezik, ezek programozhatóak és segítségükkel aktív hűtés és egyéb szenzorok működtetése lehetséges.

Rendelkezik két csatornás MIPI DSI és MIPI CSI csatlakozókkal, amik segítségével fel lehet a különböző csomópontokat szerelni képernyőkkel és kamerákkal.

A tápellátása USB-C csatlakozó segítségével történik, ez 5.1 V és 3 A-al táplálja az eszközt. [12]

Hatalmas felhasználói bázissal rendelkezik, és maga az eszköz is nagy támogatottsággal bír a szoftverek terén. Ez a felhasználók számára a könnyű kezelést segíti elő, mivel rengeteg anyag áll a rendelkezésükre, ha saját maguk szeretnének a rendszeren változtatni valamit.

Saját tesztek alapján (kettő interneten elérhető hálózati sebességmérő oldal segítségével 98) 2,4 GHz-es tartományban az átviteli sebessége (kétszer öt mérés átlag sávszélessége) ≈47 Mbps, ami nagyjából elegendő lehet, viszont, ha nagyobb teljesítmény a cél, akkor külső erősebb antennák használata elengedhetetlen.

4.4 Mikrokontrollerek

Ezek az eszközök specifikus munkákra lettek kitalálva. Jellemzően nem rendelkeznek konvencionális operációs rendszerekkel, és csak a belső memóriájukba feltöltött kódot képesek végrehajtani. Nagy előnyt jelent a kis méretük és az alacsony energiafogyasztásuk. Teljesítmény szempontjából az alacsony fogyasztás hátránnyal jár, mivel kisebb teljesítményt is eredményez. Belső maradandó memóriával ritkán rendelkeznek, ezért minden egyes újraindításkor a feltöltött kód alapján „alaphelyzetből” indulnak. Működésük közben lehetőség van konfigurálásra, viszont egy nem várt leállás a felhasználói beállításokat teljesen törli. Egyetlen maradandó konfigurációs mód a memóriájában eltárolt kód módosítása, amely nem megfelelő megoldás mindennapos alkalmazáshoz. Nagyobb szintű konfigurációt tesz lehetővé, mivel a kód egészét lehetősége van a felhasználónak módosítania, viszont ez feleslegesen bonyolítja a rendszer használatát.

Bonyolult és nagy teljesítményt igénylő rendszerek kiépítésére nem előnyös a használatuk.

5. Konklúzió

5.1 Hardver

A csomópontok megvalósításához Raspberry Pi-t alkalmasnak tűnik, mivel ez viszonylag olcsó berendezés, energiafogyasztása megfelelő és akár akkumulátorról is lehet üzemeltetni. Teljesítménye szintén elegendő a csomópontok működtetéséhez.

A csomópontok helyes működéséhez kettő antennára van szükség, ezért ezeket az eszközöket minimálisan még egy nagyobb teljesítményű Wi-Fi antennával kell bővíteni. A beépített hálózati interfész ≈50-60 Mbps átviteli sebességet képes biztosítani.

Mivel a csomópontok folyamatosan működnek, ezért az eszközök hűtését is biztosítani kell valamilyen formában.

5.2 Szoftver

A csomópontok operációs rendszerének a Linux alapú Raspberry Pi OS a legmegfelelőbb, mivel ez kifejezetten a Raspberry Pi eszközökre lett kialakítva.

A mesh hálózat működtetését és kezelését a B.A.T.M.A.N. advanced protokoll segítségével tervezem kialakítani. Ez a protokoll a megvalósítási célnak kitűzött tulajdonságok mindegyikével rendelkezik.

A konfigurációs felület webszerverének Apache2 tűnik a megfelelőnek, mivel ez egyszerűen konfigurálható, működtethető.

A weboldal alapú konfigurációs felületet érdemes használni. A weboldal struktúráját HTML, kinézetét CSS, funkcionalitását pedig JavaScript és PHP páros fogja biztosítani Jquery AJAX segítségével.

A konfigurációs felület a felhasználó interakcióit PHP segítségével fogja továbbítani a csomóponton található szkriptek felé.

Csomópontok beállítását konfigurációs felület nélkül is lehetőség van szerkeszteni, SSH-val történő távoli elérésen keresztül. Egyes csomópontokon SSH szerverek fognak működni, amik egy előre konfigurált belépési azonosítóval rendelkeznek, a felhasználónak lehetősége van publikus és privát kulcspár alapú autentikációt létesíteni az eszközökön a nagyobb biztonság érdekében.

6. Rendszerterv

6.1 Követelmények

A szakdolgozatban kialakított csomópontok központi infrastruktúra jelenléte nélküli működést is kötelesek biztosítani. Emiatt lehetőség szerint a minimális energiafogyasztás elvárt. Az egységeknek akkumulátorról is működőképesnek kell lenniük, ha az adott környezet ezt megköveteli.

Fontos, hogy a kialakított hálózaton az adatok átviteli sebessége megfelelő legyen. A hálózat csak minimális teljesítménycsökkenést tapasztalhat újabb csomópontok csatlakoztatása esetén, tehát a csomópontok nem terhelhetik le nagyon a hálózatot. A jó használhatóság érdekében a hálózatnak megfelelő adatátviteli sebességgel kell rendelkeznie. Ez akkor teljesül, ha a felhasználónak nem kell többet várnia egyes internetes szolgáltatásokra, mint általában. Ez körülbelül 50-100 Mbps-os átviteli sebességnél teljesül.

A hálózatok egyszerű beüzemelése, és bővítése fontos, mivel ez megkönnyíti a felhasználónak a rendszer használatát, és karbantartását. Könnyű hordozhatóság szempontjából is előnyös a gyors beüzemelés.

Egyes eszközök meghibásodását is tűrnie kell a rendszernek. Csomóponti hiba esetén külső beavatkozás nélkül, a rendszernek automatikusan kezelnie kell a kiesett eszköz helyettesítését. Ha elromlik egy csomópont, akkor a környezetében lévő csomópontok vehetik át a helyét. A meghibásodás és helyettesítés a rendszer teljesítményének csökkenését eredményezheti, viszont nem okozhat teljes kimaradást.

Utolsó szempont a költséghatékonyság. Mivel a kommunikációs hálózat egyik fő tulajdonsága a könnyű bővítés, hátrányos lenne, ha az egyes egységek túlságosan sokba kerülnének.

Röviden összefoglalva a szempontok:

- Alacsony energiafogyasztás,
- Megfelelő sávszélesség adatok gyors közlése érdekében,
- Könnyű beüzemelés,
- Magas hibatűrés,
- Kis helyigény.

Másodlagos szempont:

- Egyszerű integrálás más rendszerekbe.

6.2 Csomópontok felépítése

6.2.1 Hardver

A jó szoftvertámogatottság és a széleskörű elterjedtség miatt a végleges csomópontoknak Raspberry Pi 4 Model B eszközöket választottam. Ezeknek az egységeknek bármely elérhető RAM méret megfelelő.

Mivel az eszköz beépített Wi-Fi adó-vevőjének a teljesítménye nem elégséges, ezért külső antennákkal kell felszerelni, amelyek az USB 3.0 portokban helyezkednek el. A csomópontokat legalább 2 darab antennával kell felszerelni, mivel így egyszerre tudnak csatlakozni a hálózathoz és access pointként funkcionálni. Beépített Wi-Fi antennája a rendszernek egy „vészhelyzeti" megoldást biztosíthat abban az esetben, ha meghibásodna egy fő külső antenna.

Ethernet csatlakozója rendelkezni fog egy fix IP címmel, amely segítségével a csomópontok konfigurációja akkor is végrehajtható lesz, ha a szolgáltatott Wi-Fi hálózat meghibásodik. Ezen felül egyéb Ethernet csatlakozóval rendelkező eszközhöz való kapcsolat kiépítésére is alkalmas.

Az egyes csomópontokat szükséges megfelelő működés biztosítása érdekében ellátni aktív vagy passzív hűtéssel. Aktív hűtés esetén az eszköz pinjeire csatlakoztatott ventilátor megfelelő hűtést biztosít. Passzív hűtés esetén figyelembe kell venni, hogy a hűtésre használt felület megfelelő méretű legyen. Aktív és passzív hűtés kombinációjával még jobb eredmények érhetőek el.

Tápellátása attól függ, hogy a csomópontok milyen környezetben kerülnek felhasználásra. Ha rendelkezésre áll fali csatlakozó, akkor célszerű azt használni. Viszont a választott hardver alacsony fogyasztása miatt akkumulátorról is biztosítható a megfelelő tápellátás.

6.2.2 Szoftver

Operációs rendszernek hardveres kompatibilitás érdekében az ARM 64-bites Debian Linux alapú Raspberry Pi OS-t választottam ki. Ez az operációs rendszer specifikusan ehhez a SBC-hez készült, ezért nagyobb teljesítményt és jobb fogyasztást biztosít.

A hálózat szolgáltatásához a B.A.T.M.A.N. advanced nevű protokollt tervezem felhasználni, mivel ez teljesíti a szempontokat és biztosítja a hálózat teljes kapcsolatát. Előnyös, hogy a program lehetővé teszi az IPv4 és a IPv6 kommunikációt, ezáltal külső hálózatok is egyszerűen képesek a rendszerrel kapcsolatba lépni.

Access pointok létesítésére a Hostapd programot választottam, ez egyszerű és jól konfigurálható működést eredményez.

Ha szükség van DNS és DHCP szolgáltatásokra, akkor ezt a Dnsmasq program fogja biztosítani.

Webes felületét az Apache2 szolgáltatás fogja működtetni egyszerű konfigurálása és telepítés utáni azonnali használhatósága miatt.

A konfigurációs weboldalak JavaScript és PHP segítségével tervezem megvalósítani, mivel egyszerűbb a fejlesztése, kevesebb tárhelyet és erőforrást igényel. Illetve könnyebben telepíthető, mint a NodeJS segítségével történő megvalósítás. A weblap a hálózat konfigurálására szkripteket és parancsokat használ, ami a felhasználó által megadott információk alapján módosítja a B.A.T.M.A.N. advanced-et, és egyéb hálózathoz kapcsolódó beállításokat.

A konfigurációs weblap az adott csomópontról is szolgáltat információkat, ezeket egy külön információs lapon lehet elérni.

A konfigurációs weboldal igény szerint ki- és bekapcsolható. Ez az erőforrások spórolása és a teljesítmény növelése érdekében jöhet szóba. Könnyű kezelése érdekében a teljes felületet egy Systemd szolgáltatás segítségével lehet be-, illetve kikapcsolni.

Ahhoz, hogy weboldal nélkül is képesek legyünk a hálózatot módosítani, illetve a weboldalt visszakapcsolni, egy SSH szerver működik a csomóponton. Ezáltal a távoli biztonságos elérés biztosított, és a felhasználónak is lehetősége van az egyes csomópontok magas szintű konfigurálására.

7. Megvalósítás

A rendszert a terveknek megfelelően sikerült megvalósítanom. A csomópontok modularitása végett egy plugin rendszert is kialakítottam, külön saját fejlesztésű plugin kezelő programmal.

A fejezetben az általam fejlesztett rendszer és egyéb ehhez kapcsolódó szakdolgozat során megvalósított szoftver kerül részletesebb bemutatásra.

7.1 Csomópont

Ebben a fejezetben az megvalósított csomópont felépítését és működését mutatom be.

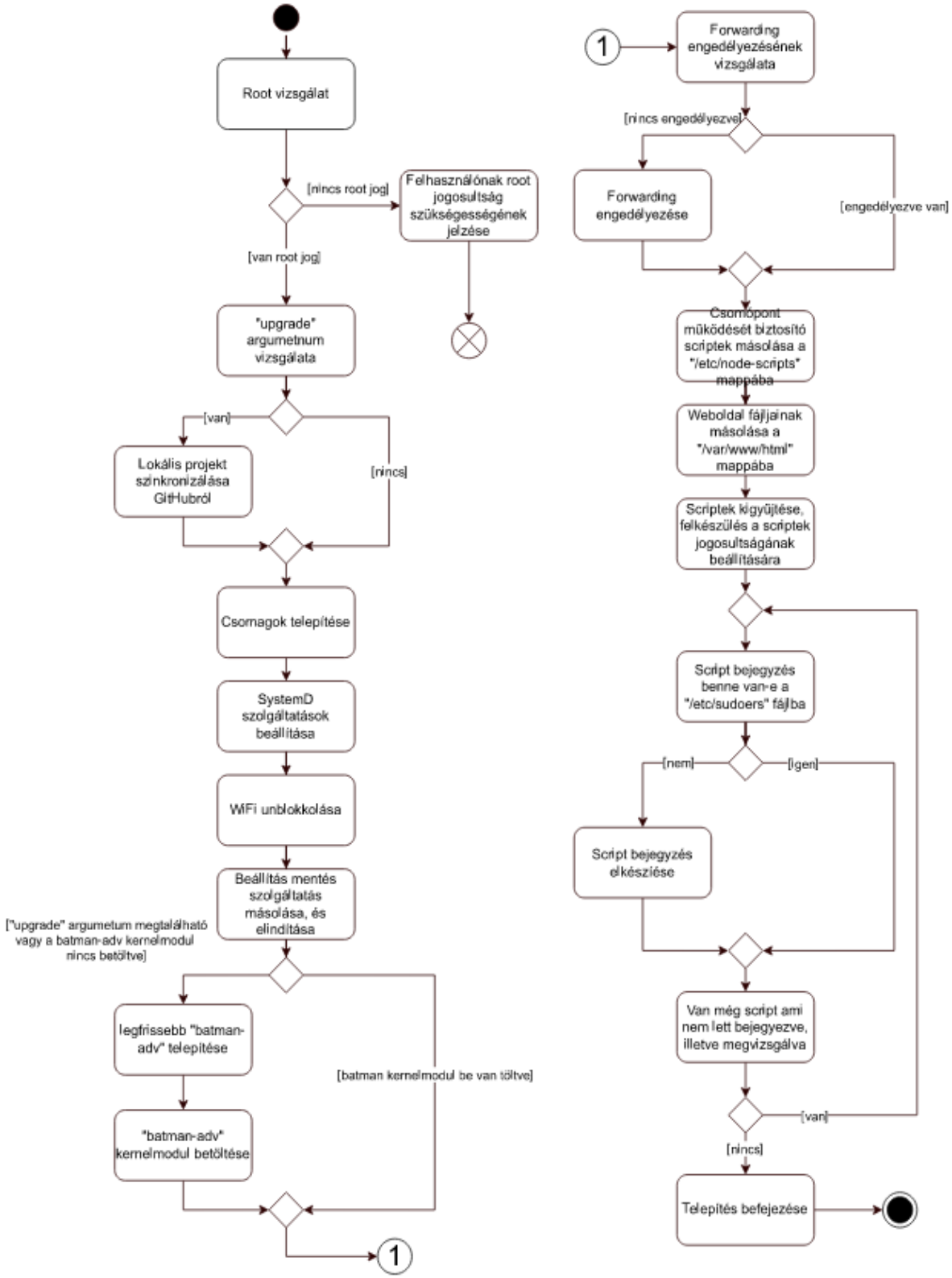
7.1.1 Telepítés

A telepítés legfőbb szempontja az egyszerűség volt megtervezésem során. Célom volt, hogy a felhasználó minél gyorsabban tudja a rendszerét működésbe helyezni. Telepítés után a rendszer webes konfigurációs felülete egyből elérhető, és a hálózat további lépések nélkül konfigurálható.

A telepítést egy Bash szkript végzi. Név szerint ez a projekt gyökerében található „setup.sh” fájl. A telepítés futtatásához szükséges, hogy a felhasználónak legyen root jogosultsága, mivel olyan műveletek hajtódnak végre, amikhez ez előfeltétel.

A telepítő szkriptet úgy alakítottam ki, hogy rendelkezzen egy frissítés funkcióval is. Ezt a felhasználónak külön kell specifikálnia az „upgrade” argumentummal. Ha ezt az argumentumot megadja, akkor a telepítő szinkronizálja a projektet a GitHubomon lévő verzióval.

A telepítés folyamán a telepítő a megfelelő helyre másolja a szakdolgozatban megvalósított rendszer fájljait.



11. ábra: Telepítés folyamatdiagramja.

7.2 Konfigurációs felület

Konfigurációs felület a rendszertervnek megfelelően alkottam meg, tehát HTML alapú és CSS biztosítja a kinézetét. A felhasználói felület kommunikációját a szerver oldallal JavaScript és PHP segítségével valósítottam meg. Az összedolgozásukat Jquery Ajax biztosítja.

A felület webböngészőn keresztül érhető el. Elérése a csomópont IP címével történik.

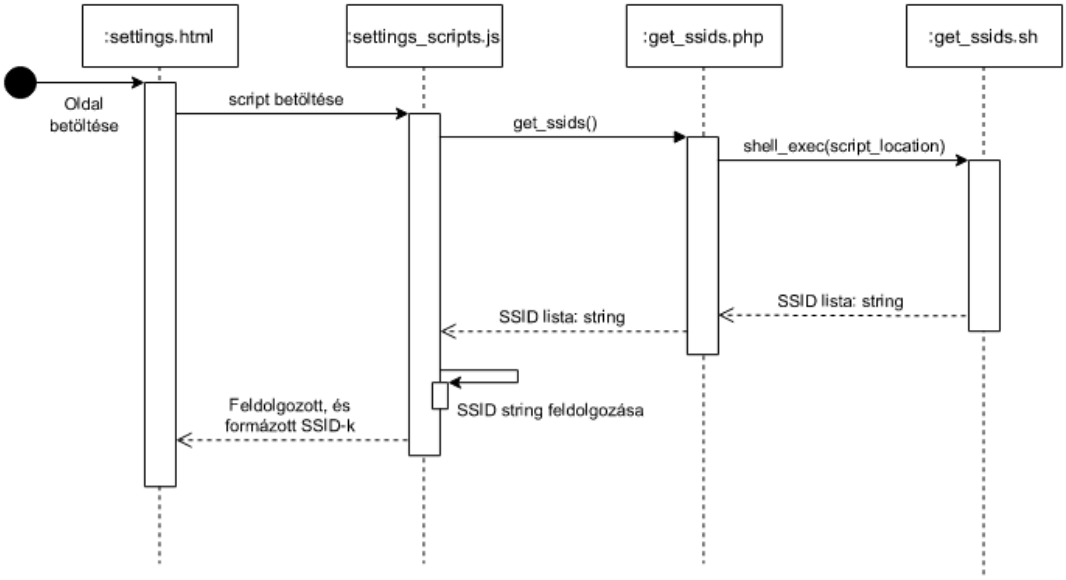
7.2.1 Felépítés

A csomópontok konfigurálásában több réteg vesz részt. Ezek egymásra épülnek, és egymással kommunikálnak. Úgy alakítottam ki ezeket, hogy mindegyik csak az alatta lévőt érheti el, ezzel biztosítva a rétegek szabad módosítását.

A rendszert a Raspberry OS-re fejlesztettem. Mivel előfordulhat, hogy más Linux disztribúciók más működésekkel rendelkezne, a rétegek szabad cserélése miatt a rendszer portolása egyszerű, mert csak egy adott réteg működését kell módosítani.

A rendszer rétegződését a következőképpen lehet szemléltetni:

1. Megjelenítési réteg (legfelső réteg): A felhasználói felület rétege. Ebben a rétegben találhatóak a HTML, illetve a CSS fájlok. Célja az oldal struktúrájának megadása, és kinézetének formálása.
2. Akció réteg: Ebben a rétegben szereplő elemek közvetlenül a „megjelenítési” rétegből hívódnak meg. Ezek lehetnek JavaScript, illetve PHP fájlok. Céljuk a felső rétegből bejövő adatok kezelése, feldolgozása, illetve szükség esetén ezek továbbítása az alsó rétegnek. Valamint a megjelenített adatok frissítése.
3. Rendszerközeli réteg: Ez a legalsó réteg. Ezt Bash szkriptek és olyan PHP fájlok alkotják, melyek rendszerparancsokat képesek futtatni. A szkriptek jogosultságára ügyelni kell, mivel biztonsági problémákat okozhatnak, csak a szükséges szkripteknek szabad ilyen jogokkal rendelkeznie. Célja a rendszer működésének módosítása, illetve adatok kinyerése a rendszerből.



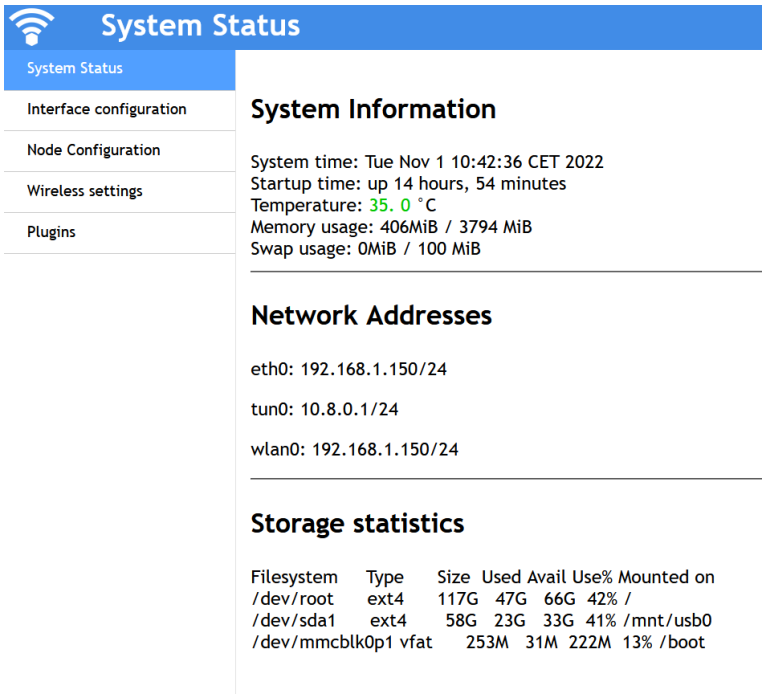
12. ábra: Rétegződési példa, SSID-k megjelenítésének szekvencia diagramja.

A felhasználói felületet több részre bontottam, elkülönítve a különböző funkciókat. Ezek a következőek:

1. Rendszerállapot
2. Interfész konfiguráció
3. Csomópont konfiguráció
4. Vezeték nélküli konfiguráció
5. Pluginok

7.2.2 Rendszerállapot

A felhasználó valós idejű adatokat lát a csomópontról ezen az oldalon. Létrehozása közben célom az volt, hogy egy helyen minél többmindent lehessen megtudni a csomópontról.



13. ábra: Rendszer állapot oldal. Rendszerről biztosít adatokat a felhasználónak.

Informatív szerepet biztosít. A rendszer beállított idejéről, az indítás óta eltelt időről, a processzor hőmérsékletéről és a jelenleg használt memória mennyiségéről szolgál adatokat.

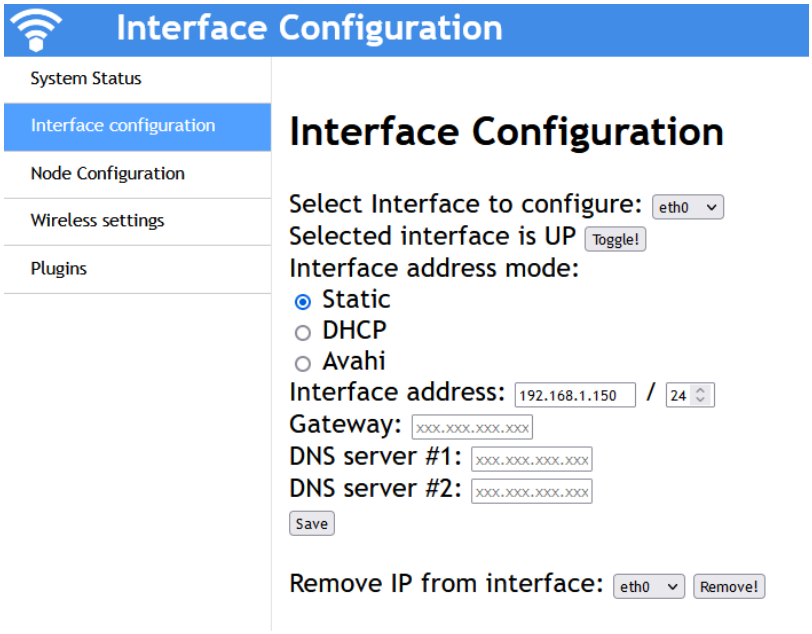
Biztosít hálózati információkat is, mint például a jelenleg elérhető interfészek, és ha vannak akkor ezek címei.

A rendszer tárhelyéről is ad információkat ez az oldal. Ezt százalékos formában teszi meg. Ez az információ fontos lehet, ha például megosztott adattárolási szerepet tölt be a csomópont.

Az az oldalt úgy alakítottam ki hogy az itt található adatok sűrűn frissülnek, hogy mindig az aktuális állapotát mutassa a csomópontnak.

7.2.3 Interfész konfiguráció

Az oldal lehetőséget nyújt a felhasználónak az interfészek konfigurálására. Az interfészek egy lenyíló menüből választhatóak ki. Lehetőség van statikus, DHCP által osztott és Avahi rendszer által nyújtott címeket használni. Illetve egy külön opció nyílik az adott interfész címének törlésére. A felhasználó ezen a felületen tudja fel- és lekapcsolni az adott interfészeket. Az oldalt úgy valósítottam meg, hogy dinamikusan figyelje, hogy éppen melyik címezési módszer van kiválasztva, és eszerint változtassa a láthatóságát az interfész beállításoknak.

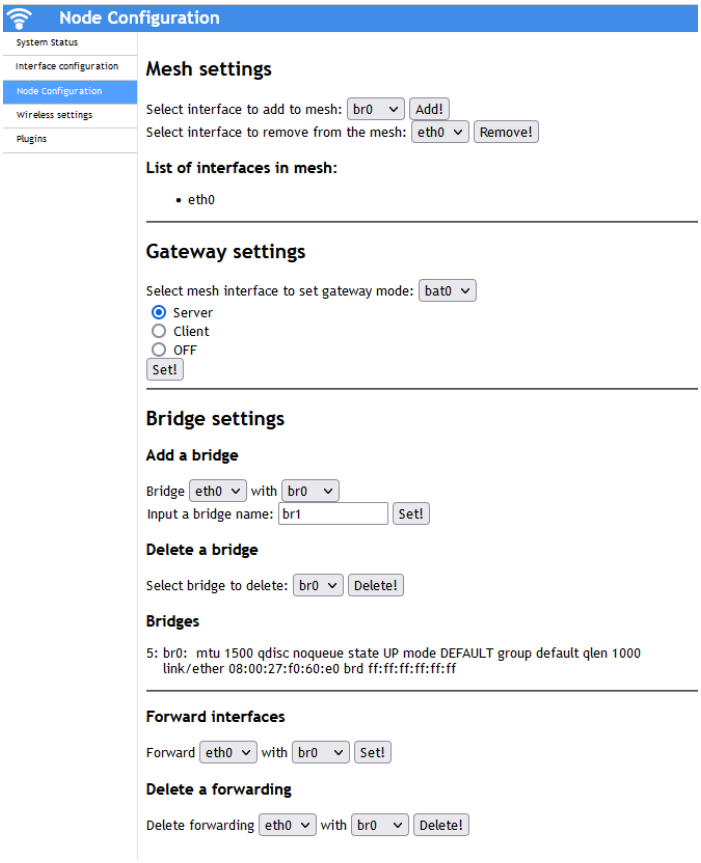


14. ábra: Interfész konfigurációs oldal.

7.2.4 Csomópont konfiguráció

Ez az oldal biztosítja a felhasználó számára a csomópont hálózati beállításának lehetőségeit, illetve a batman-adv beállításai itt találhatóak. Ezek használata későbbi fejezetben lesz részletezve.

Az oldal információt is nyújt a csomópont hálózati állapotáról. A lenyíló menükre fejlesztés során ügyeltem, hogy mindig csak az éppen aktuális interfészek azonosítóját tartalmazza.

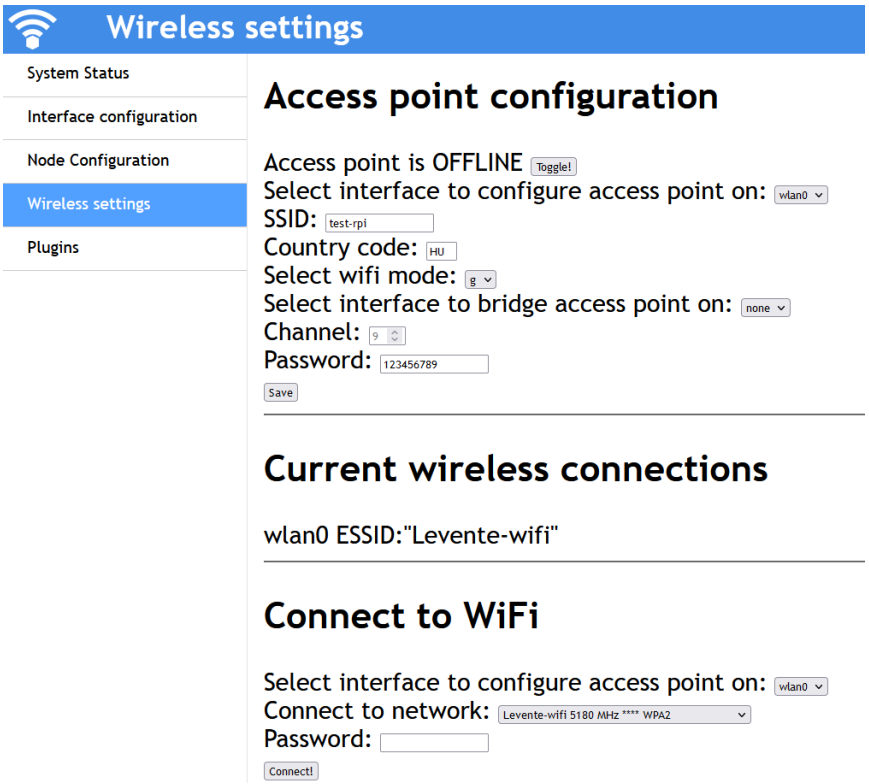


15. ábra: Csomópont konfigurációs felület.

7.2.5 Vezeték nélküli beállítások

Az oldalon a vezeték nélküli konfigurációk találhatóak, mint például a hotspot és a WiFi csatlakozás beállításai. Kialakítottam egy lenyíló menüt az oldalon, amely az elérhető vezetéknélküli hálózatokat megjeleníti. A lenyitható menüben látszik az elérhető hálózat SSID-je, működési frekvenciája, erőssége és titkosításának típusa.

Az oldalon megjelenik a jelenlegi WiFi kapcsolat is.



16. ábra: Vezeték nélküli beállítások oldala.

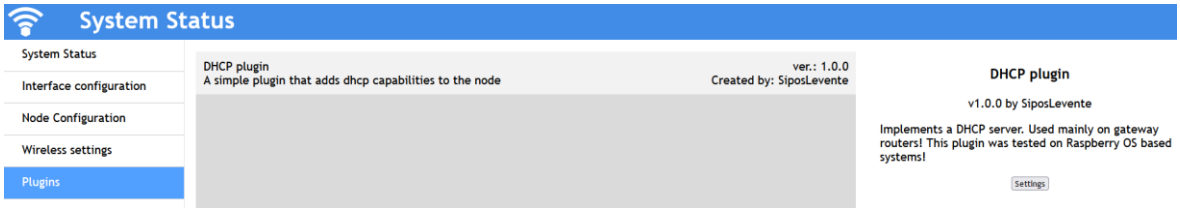
7.2.6 Pluginok

Az oldalon megtalálhatók a telepített csomóponti pluginok. A pluginok a rendszer funkcionalitásának bővítését biztosítják. A pluginok részletesebb bemutatása későbbi fejezetben szerepel.

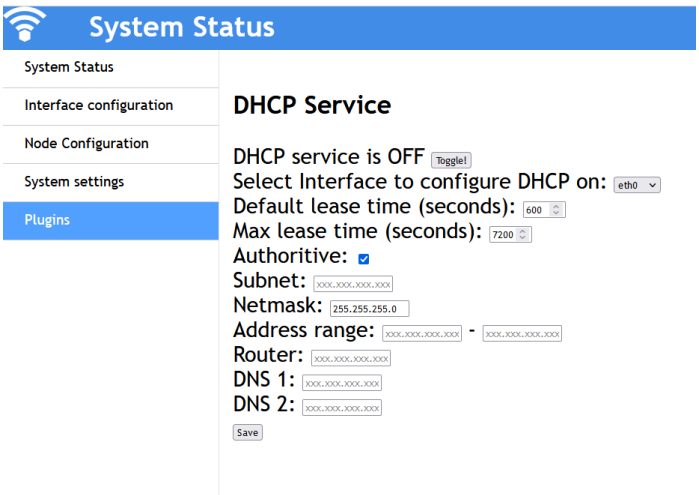
Az oldalt úgy alakítottam ki, hogy egy görgethető listában szerepeljenek a pluginok, rájuk kattintva az oldal bal részén megjelenik a részletesebb leírásuk, illetve, ha a plugin is támogatja, akkor innen lehet elérni a bővítmények beállításait.

Az oldal pár másodpercenként frissíti a pluginok listáját, tehát az oldal frissítése nélkül elérhetőek a frissen telepített pluginok.

Pluginok kezelése és telepítése későbbi fejezetben kerül részletezésre.



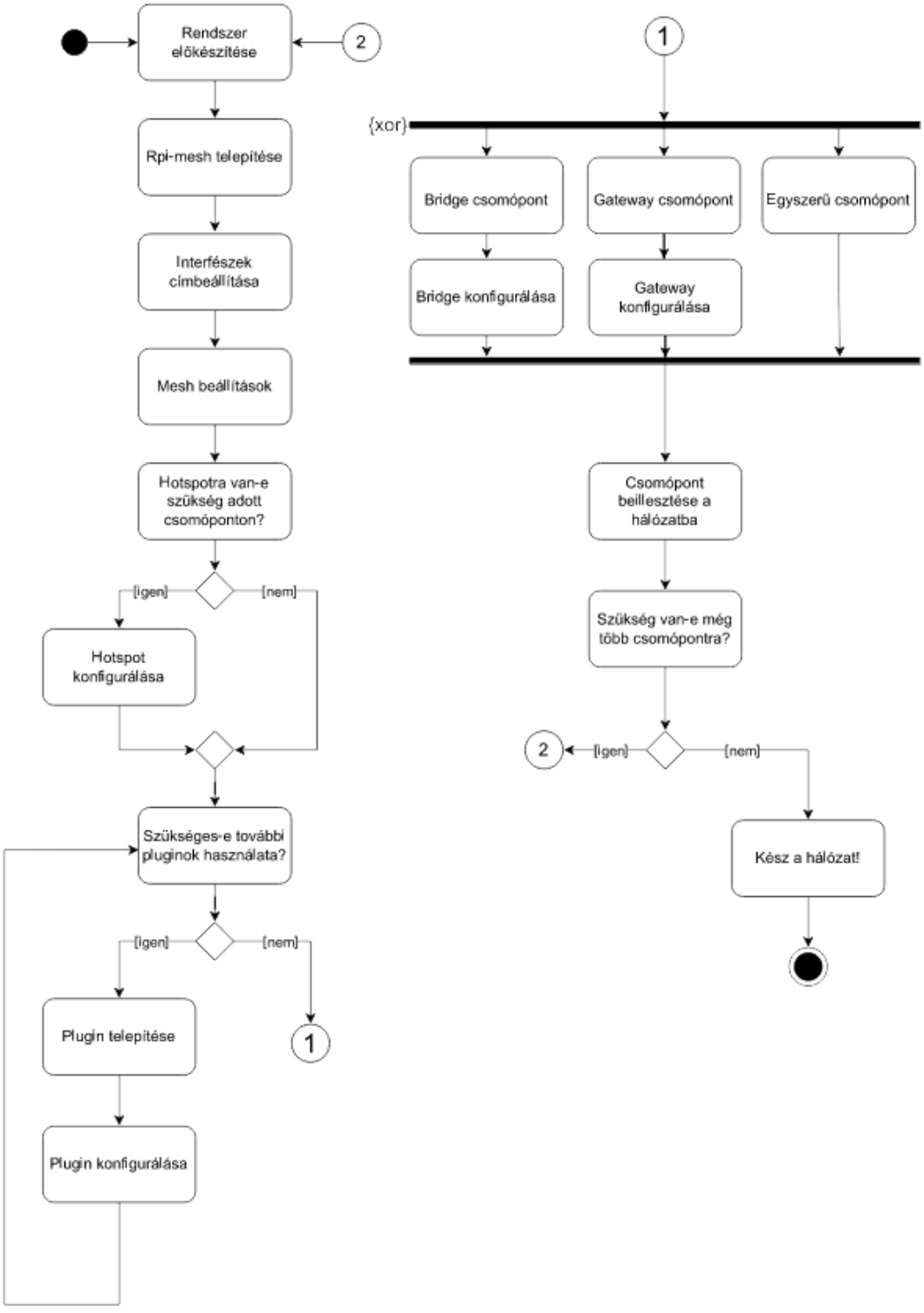
18. ábra: Pluginok oldal, egy betöltött DHCP pluginnal.



17. ábra: DHCP példa plugin konfigurációs felülete.

7.2.7 Használat

A fejezet folyamatmodellel szemléltetem egy egyszerű mesh hálózat összerakását.



19. ábra: Hálózat kialakításának folyamatmodellje.

7.3 Maradandó beállítások

Mivel nem mindegyik program rendelkezik maradandó beállítási fájlokkal, ezért meg kellett oldanom, hogy a rendszer újraindítása után is megmaradjanak ezek beállításai.

A megoldást egy SystemD alapú saját készítésű szolgáltatás nyújtotta. Ez a szolgáltatás minden induláskor egy szkriptet futtat. A szkirpt tartalma bizonyos nem maradandó beállítások változtatásakor frissül, olyan módon, hogy ezek újraindítás esetén is a beállított állapotba kerüljenek.

Ezek az indító szkript módosításokat a konfigurációs felület által használt rendszerközeli bash szkriptekbe építettem be.

7.4 Hálózat

A rendszerterv szerint a hálózati csomópontok közötti decentralizált mesh kapcsolatot a batman-adv szolgáltatás segítségével valósítottam meg. Ez a szolgáltatás lehetővé teszi az egyszerű konfigurációt, mivel az interfészek mesh hálózatba léptetésén kívül egy alap csomópontnak nincs más beállításra szüksége. Batman-advanced segítségével a hálózat úgy tekinthető, mintha egyetlen virtuális „switch”-en lenne.

A hálózatok öngyógyító tulajdonságát a csomópontok WiFi roaming képessége teszi lehetővé. Ez lehetővé teszi az eszközöknek, hogy amennyiben az éppen csatlakozott másik csomópont meghibásodik, esetleg tönkremegy, akadályok nélkül átcsatlakozzanak egy másik ugyanolyan tulajdonságokkal rendelkező hálózathoz. Biztonsági megvalósítása abban rejlik, hogy csakis akkor csatlakozik a csomópont a másik hálózathoz, ha az adott hálózat ugyanazzal a névvel és titkosítási beállításokkal rendelkezik.

7.5 Pluginok

A pluginok konfigurációs felületét hasonlóan alakítottam ki, mint a többi oldalt. Rendelkeznek opcionálisan egy telepítő szkripttel. Ez a szkript telepíti a szükséges csomagokat, és a plugin fájljait a megfelelő helyekre mozgatja.

A pluginok a weboldal mappájában a „plugins” mappában találhatóak, és ide is kell őket telepíteni.

Mivel a pluginok is képesek a csomópont rendszerének módosítására, ügyelni kell ezek biztonságára. A pluginok biztonságát egy központi plugintárhely biztosíthatja, amelyben csak olyan bejegyzések szerepelnek, amelyek biztonsági szempontokból át lettek vizsgálva. A pluginkezelő és a központi tárhely későbbi fejezetben kerül részletezésre.

Pluginokonat úgy alakítottam ki, hogy ezeken keresztül a rendszer bármilyen tulajdonsággal, szolgáltatással és működéssel felruházható. Egyetlen követelmény, hogy a plugin felépítése kövesse a specifikus felépítési szabályokat, és rendelkezzen egy konfigurációs felülettel abban az esetben, hogyha ez szükséges.

7.5.1 Telepítés

A pluginok telepítésének két módját határoztam meg:

1. Manuálisan,
2. Pluginkezelő segítségével.

Mindkét mód ugyan azt eredményezi, viszont a pluginkezelő használatával az installálás gyorsabb és biztonságosabb.

Manuális installáció esetén, ha a plugin konvencionális felépítéssel rendelkezik, akkor a telepítő fájljai egy „setup_scripts” nevű mappában találhatóak. Ez a mappa tartalmazza a plugin „setup.sh” fájlját, amely a telepítést végzi, és egyéb rendszerfájlokat, amik a plugin működéséhez szükséges.

A plugin manuális telepítésének elindításához a plugin „setup.sh” szkriptet kell futtatni. Pluginkezelő ezt a „setup.sh” fájlt automatikusan futtatja.

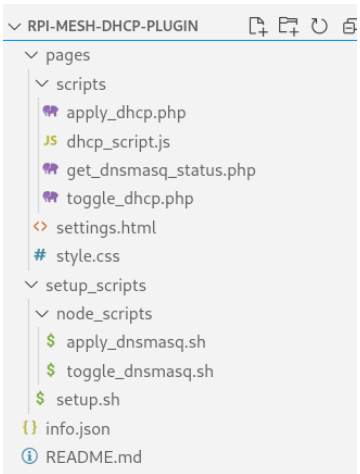
7.5.2 Felépítés

A pluginok felépítését úgy határoztam, hogy a megfelelő működés érdekében egy adott sémához kell illeszkedniük. A séma 3 fő részből áll:

1. oldalak mappa,
2. installációs mappa,
3. információs fájl.

Az oldalak mappa tartalmazza a megjelenítéshez és oldal funkcionalitásához szükséges fájlokat, tehát itt találhatóak a „megjelenítési” és „akció” réteg fájljai, esetlegesen pár „rendszerközeli” PHP fájlt.

Az installációs mappa tartalmazza a telepítéshez szükséges scripteket, illetve a plugin „rendszerközeli” rétegének a fájljait. Ezek a fájlok a telepítés során a rendszer számára megfelelő mappába másolódnak.



20. ábra: Plugin felépítése.

7.5.3 Infos.json

Ez a fájl egy közönséges JSON fájl. Célom fejlesztés közben a fájllal a plugin adatainak tárolása volt. Ennek segítségével töltődik fel a pluginok konfigurációs oldala adatokkal. Más funkciója a pluginok könnyebb beazonosítása.

```
{
"pluginName": "DHCP plugin",
"version": "1.0.0",
"author": "SiposLevente",
"short_description":"A simple plugin that adds dhcp capabilities to the node",
"long_description":"Implements a DHCP server. Used mainly on gateway routers!
}
```

21. ábra: infos.json felépítése.

Jövőbeli fejlesztési lehetőséget egy publikus kulcs mező, amely a biztonság növelését eredményezheti.

7.6 Pluginkezelő

A pluginkezelőt a plugin telepítésének és kezelésének megkönnyítése érdekében hoztam létre. Rust programozási nyelvben készítettem ezt a programot a gyors működés, és hibák könny kiküszöbölése érdekében.

A program egy parancssori alkalmazás. A felhasználó argumentumok segítségével tudja a kezelő működését irányítani. Pluginokat egy központi tárolóban tartja nyilván. Ennek a tartalma lokálisan van tárolva, viszont egy távoli központi tárhelyből frissíthető, jelen esetben ez egy GitHubomon található publikus git tároló. A felhasználónak lehetősége van saját tárolókat is létrehozni, ehhez a tárhelyek felépítésének sémáját kell követnie. Ez a séma későbbi fejezeteben lesz kifejtve.

A pluginkezelő futtatásához adminisztrációs jogosultság szükséges.

7.6.1 Argumetnumok

Hat argumentumot hoztam létre a pluginkezelő működtetéséhez:

1. „install”: installálni lehet vele a tárolókban található pluginokat. Ha a pluginban található egy „setup.sh” script akkor a program azt is lefuttatja. Használata: „sudo rpi-mesh-plugin-manager install [PLUGIN NÉV]”
2. „update”: Frissíti a tárolókat, a bennük megadott elérési útvonal alapján. Használata: „sudo rpi-mesh-plugin-manager update”
3. „upgrade”: Frissíti a specifikált plugint. Ha nincs megadva plugin név akkor mindegyik plugint frissíti. Használata: „sudo rpi-mesh-plugin-manager upgrade [PLUGIN NÉV]”
4. „uninstall”: Uninstallálja a megadott plugint. Használata: „sudo rpi-mesh-plugin-manager uninstall [PLUGIN NÉV]”
5. „list”: Kilistázza a tárolókban elérhető összes plugint. Használata: „sudo rpi-mesh-plugin-manager list”.

6. „help”: Kilistázza az összes opciót és használati módot. A help parancs jelenik meg minden egyes olyan argumentum beütése esetén, ami nincs előre specifikálva. Használat: „sudo rpi-mesh-plugin-manager help”

7.6.2 Működése

A pluginkezelőt úgy állítottam be, hogy alapértelmezetten a „/etc/rpi-mesh-plugin-manager/” mappában keresi a fájljait és a lokális tárolókat.

Itt található elemek:

1. „config.conf” fájl
2. „plugin.repo” fájl
3. „repos” mappa
4. „.installed” rejtett fájl

Ezek felépítése későbbi fejezetekben kerül részletezésre.

A beállításokat, (mint például telepítési célmappa, tárolók helye), a „config.conf” fájl tárolja. Ezt a felhasználó szabadon módosíthatja, amennyiben van hozzáférése.

A program minden induláskor betölti a memóriába az elérhető pluginokat a „plugin.repo” fájlból és a „repos” mappában található tárolók tartalmából, amennyiben a specifikált működés ezt megkívánja.

7.6.3 config.conf

Könnyen módosítható szöveg alapú konfigurációs fájl. Minden sor egy külön beállítást reprezentál. Úgy alakítottam ezt ki, hogy kialakítása és feldolgozása miatt könnyen bővíthető legyen.

```
installed_cache_location: .installed
plugin_folder_location: plugins
plugin_repo_location: plugins.repo
```

22. ábra: Alapértelmezett tartalom.

7.6.4 plugin.repo

A pluginkezelő alapértelmezett „hivatalos” tárolója. Külön kezeltem a többi tárolótól. A tartalmának bővítése nem támogatott, frissítéskor minden fájlban történt módosítás felülíródik, mivel ez a tároló csak a központilag elfogadott és kivizsgált pluginoknak van fenntartva.

7.6.5 repos

Ez a mappa tartalmazza a felhasználók által definiált tárolókat. Felhasználása ugyanúgy történik, mint a „hivatalos” tárolóé. Eltérés abban van, hogy ennek a tartalma frissítéskor nem törlődik, hanem kibővül.

A mappa akármennyi tárolót tartalmazhat. Ezeknek a létrehozott saját tárolóknak célszerű a „.repo” kiterjesztést használnia.

7.6.6 .installed

A fejlesztett pluginkezelő ebben tartja számon a telepített pluginokat. Úgy alakítottam ezt ki, hogy tartalma módosuljon plugin telepítésekor és törlésekor. Ez egy egyszerű szöveges fájl, amelyben a pluginok fel vannak sorolva.

Telepítés folyamán, ha sikeresen települ egy alkalmazás, akkor ez az „.installed” fájlba beíródik. Uninstalláció esetén ebből a fájlból az adott bejegyzés törlődik.

7.6.7 Tárolók felépítése

A konvencionális tároló felépítését két fő részre bontottam, ezek a részek a távoli frissítési forrást jelképező rész, és a plugindefiníciós rész.

A távoli forrást jelző rész a „remote” kulcsszóval ellátott rész. A pluginkezelő frissítéskor innen szinkronizálja a tartalmát a fájlnak.

A plugindefiníciós rész a fájl távoli forrását jelző résztől eltérő összes többi rész.

Egy fájlban akármennyi pluginbejegyzés lehet. A pluginbejegyzések kezdetét szögletes zárójellel körbefogott plugin név kezdi. Ez a plugin név határozza meg, hogy a kezelőből a definiált plugin hogyan érthető el. Az „enabled” kulcsszó egy bool típusú mezőt reprezentál, amely jelképezi, hogy az adott bejegyzést a rendszer figyelembe vegye-e. Típusát a „type” kulcsszó reprezentálja.

Ennek fajtái:

1. collection: Más tárolóra mutató bejegyzést reprezentál.
2. repo: Távoli git repository, telepítés során tárolót „clone”-oz a célmappába.
3. local: Helyi mappát reprezentál, plugin telepítésekor másolás történik.

A „location” kulcsszó a plugin helyét jelöli, tehát a telepítés során a plugin fájljait innen szerzi be a kezelő.

```
remote=https://raw.githubusercontent.com/SiposLevente/rpi-mesh-plugin-manager/main/plugins.repo

[rpi-mesh-dhcp-plugin]
enabled=true
type=repo
location=https://github.com/SiposLevente/rpi-mesh-dhcp-plugin
```

23. ábra: Tároló felépítése.

7.6.8 Pluginok tárolása a memóriában

A Pluginokat egy hasítótáblában helyeztem el a pluginkezelő programon belül, mivel több felhasználó által definiált tárhely kerülhet működés során felhasználásra, amely esetlegesen ugyanazt a plugint tartalmazza. Másik előnye a pluginok gyors elérése, mivel a hasítótábla kialakítása lehetővé teszi, hogy amennyiben a plugin megtalálható a memóriában, azt egy lépéssel elérjük.

7.7 Tapasztalatok

A szakdolgozat megvalósításához sok kis különálló részt kellett létrehoznom. A rendszert ezeknek a kis részeknek az együttműködése alakítja ki. Tagoltsága miatt

elengedhetetlen volt ezek követése és számontartása. Erre egy, a rendszer felépítését reprezentáló diagramot hoztam létre. (Lásd 1. melléklet) Ez segített a fejlesztésben, és nyomon tudtam követni mit implementáltam már, és mit kell még megvalósítani.

A projekt kis részekre darabolása segítette a fejlesztést, mivel minden egységnek csak egy részfeladatot kellett ellátnia. Ez elősegítette az egyes részek újrafelhasználását is. A hibakeresést is gyorsította, mivel pontosan meg lehetett mondani, hogy a nem várt működést eredményező rész melyik fájlban található. A fejlesztés menetét kis mértékben lassította, mivel a rengeteg fájl közül meg kellett találni a szerkeszteni kívántat.

A szakdolgozat megvalósítása közben git verziókezelőt használtam fel. Ez egyrészt biztonsági funkciót látott el, esetleges adatvesztést lehetett kikerülni vele. Másik hasznos funkciója a tesztelés gyorsítása, mivel nem kellett mindig kézzel az eszközökre másolni a telepítő fájljait, hanem elegendő volt egy git „clone” funkcióját használni a távoli tároló letöltéséhez. Távoli git tárolónak a GitHub rendszerét használtam. A verziókövetés funkciója is hasznos volt, mivel így egy esetleges nehezen kijavítható végzetes hiba esetén egyszerűen vissza lehet állni egy régebbi állapotra.

A fejlesztés menete gördülékenyen zajlott, nagy akadályba nem ütköztem a megvalósítás során.

A kisebb méretű funkcionalitást validáló folyamatok vizsgálata eleinte nehézkes volt a kevés rendelkezésre álló eszköz miatt. De miután ezeket a teszteseteket kivizsgáltam, virtuális környezetben a nagyobb rendszerek tesztelését egyszerűen meg tudtam valósítani.

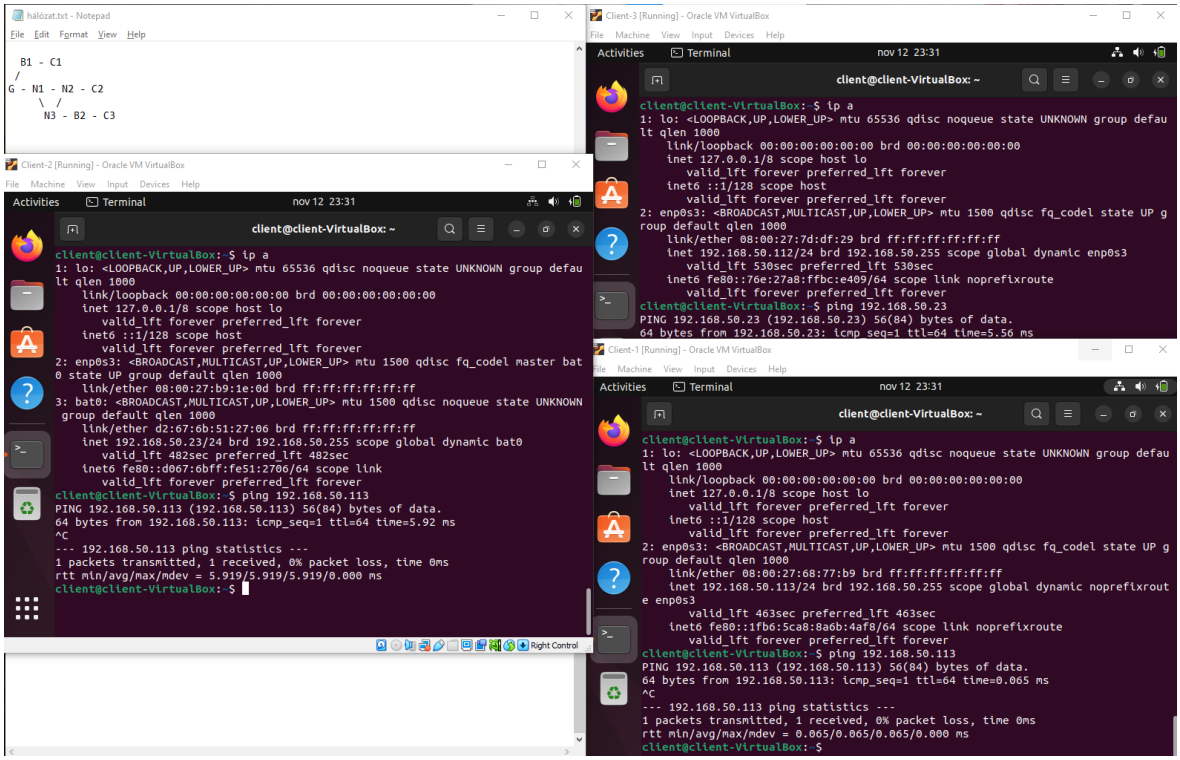
8. Megvalósított hálózat bemutatása

A fejezetben egy kisebb megvalósított hálózat kerül bemutatásra. Mivel virtuális környezetben készítettem el a hálózatot, ezért csak vezetékes kapcsolatokat lehetett alkalmaznom csomópontok összeköttetésére.

A kisebb funkcionalitást verifikáló tesztek a hálózat újracsatlakozási képességét igazoltam, ezért a virtuális megvalósítás végezhető vezetékes kapcsolatok implementálásával is.

8.1 Hálózat bemutatása

A hálózat bemutatása során egy kisebb céges hálózatot mutatok be. Mind a batman-adv hálózati képességekkel rendelkező eszközök és az ezzel nem felszerelt gépek képesek a hálózathoz csatlakozni. A hálózat számára internetelérés is biztosított. Mindegyik eszköz eléri a másikat (Lásd:



27 ábra).

8.2 Hálózat felépítése

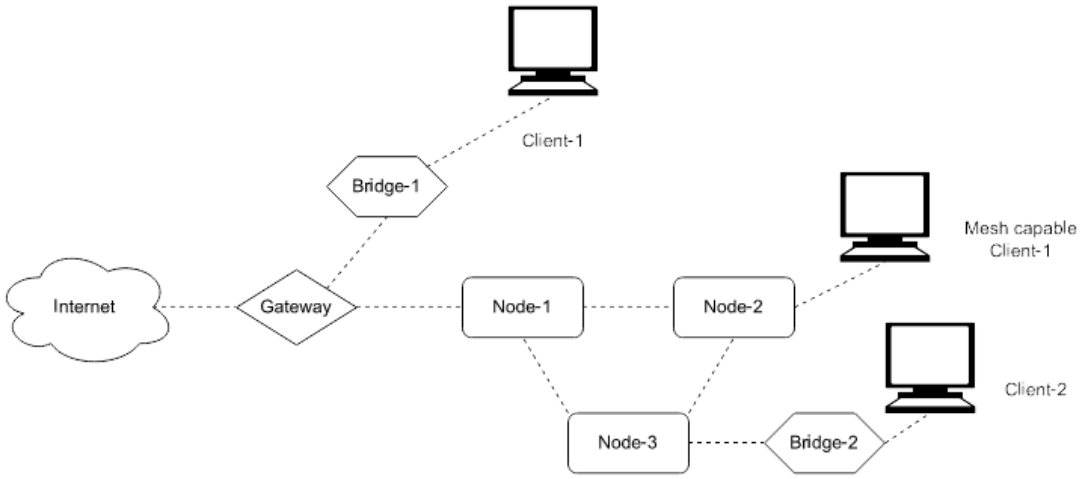
A hálózatban kilenc eszköz található. Három-három kliens berendezés, és közönséges csomópont, kettő „bridge” funkcióval ellátott csomópont, egy „gateway” csomópont.

A gateway segítségével elérhető a külvilág a belső hálózatból, tehát ez az eszköz biztosít internetelérést.

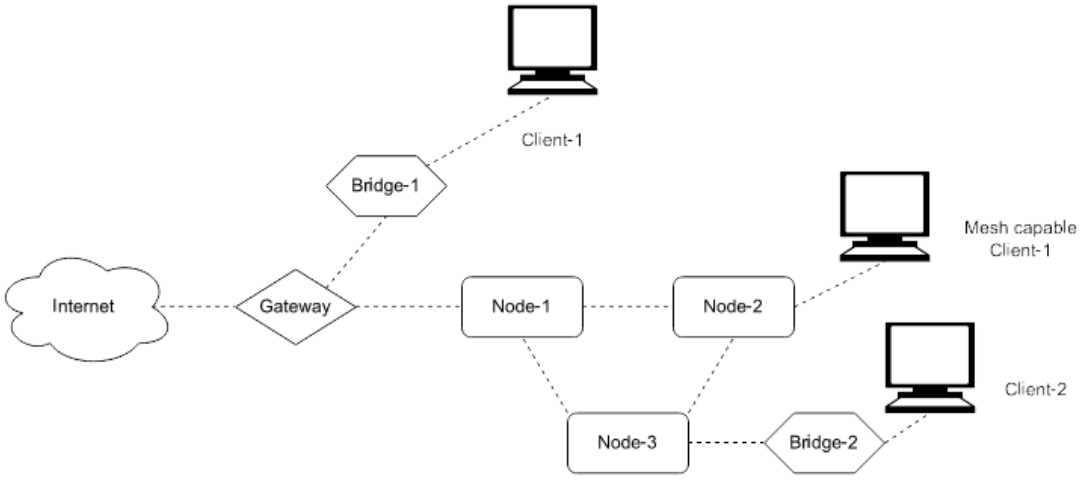
A bridge más hálózatokba nyújt hozzáférést. Ebben az esetben olyan klienseket csatlakoztat, amelyek nem rendelkeznek batman-adv hálózatba való kapcsolódási képességekkel.

A közönséges csomópontok jelenleg csak a csatlakoztatás biztosítják az egyes hálózatbéli komponensek között, de pluginok segítségével fontos részei lehetnek a hálózatnak, például adatbázisokat és egyéb szolgáltatásokat biztosíthatnak. Természetesen a csomópontok teljesítményének megfelelő feladatokat kell az eszközökre bízni.

A hálózat felépítése (lásd



24. ábra: Hálózat illusztrációja.



24 ábra) viszonylag egyszerű. Konvencionális eszközökkel történő felépítés során a hurok miatt a csomagok kezelésének irányításával foglalkozni kellene. Erre a megvalósított rendszerben nincs szükség, mivel a csomópontok automatikusan kezelik az ilyen eseteket.

8.3 Hálózat konfigurációja

A szakdolgozatban létrehozott rendszernek köszönhetően a hálózat konfigurálását gyorsan tudtam elvégezni. Elegendő volt típusonként egyet-egyet beállítani, majd ezeket a beállításokat, átmásolni a többi ugyanilyen típusú eszközre. Az átmásolás során az indulást kezelő „start.sh” szkriptet és esetlegesen a módosított interfészek beállítását kezelő konfigurációkat volt szükséges egyik eszközről a másikra helyezni. A teljes kiépítés emiatt ≈20-30 percet vett igénybe.

8.3.1 Gateway eszköz

A közönséges csomópontoktól abban tér el, hogy képes adatokat továbbítani az egyik interfészéről a másikra. Ez a továbbítás „forwarding”, azaz továbbítási funkció segítségével létesül. A továbbítás az internet felé található interfész és a virtuális batman-adv által szolgáltatott interfész között helyezkedik el.

Továbbítási funkció mellett DHCP szolgáltatást is nyújt a hálózat számára, tehát ennek az eszköznek a segítségével történik a hálózatban a címosztás.

8.3.2 Bridge eszközök

A csomópont nevéből eredően „bridge” beállításokkal rendelkeznek. A virtuális batman-adv interfész kerül „bridge”-elésre a külső hálózat felé néző interfésszel. A „bridge” megvalósításhoz szükséges egy „bridge” interfész létrehozása.

8.3.3 Csomóponti eszközök

Csomóponti eszközök biztosítják a hálózat résztvevői közötti kapcsolatot. Ezekhez minden megfelelő biztonsági adatokkal rendelkező, batman-adv rendszert implementáló eszköz csatlakozni tud.

Fő beállításai annyiból állnak, hogy az általunk kiválasztott interfészeket beléptetjük a batman-adv hálózatba.

Jelenlegi megvalósításban nem végeznek plusz munkát, viszont valós környezetekben szolgáltatások futhatnak rajtuk.

8.4 Konklúzió

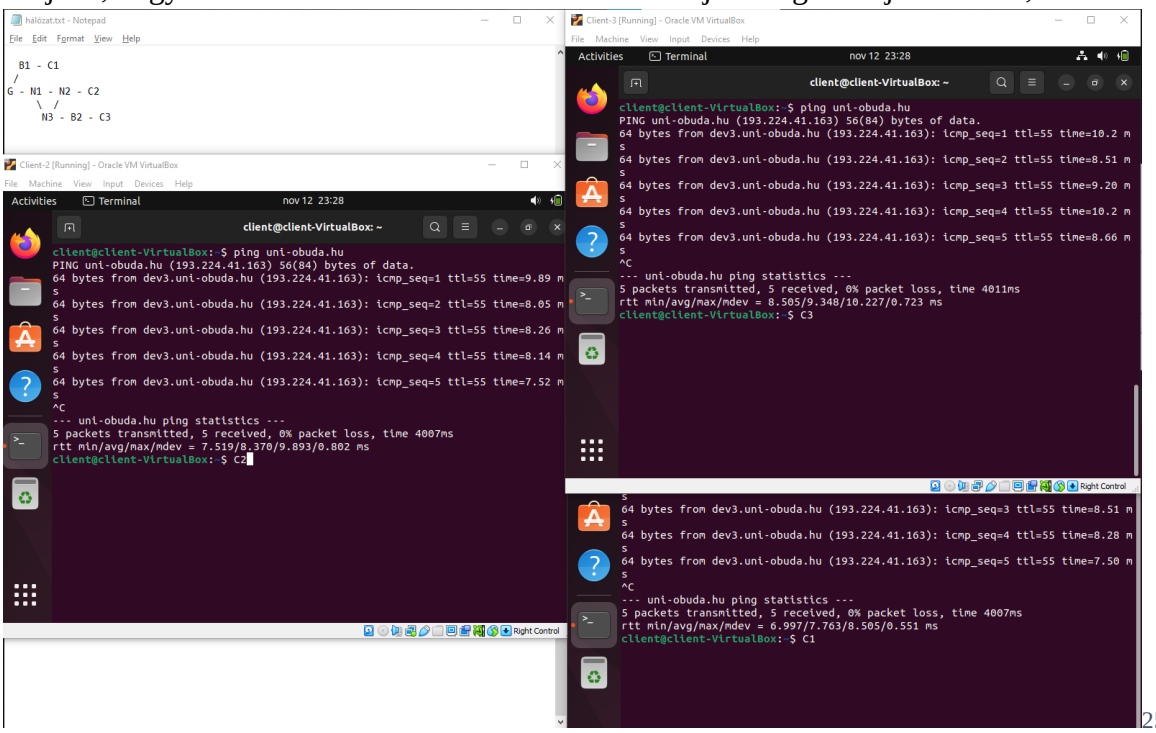
A hálózat kialakítása egyszerű volt, konfigurálása webes felhasználói felület segítségével gyorsan történt.

A kialakított hálózat átviteli sebessége megfelelő, viszont ügyelni kell arra, hogy minél több csomópont mögött van egy eszköz, annál nagyobb lesz a hálózat késleltetési ideje. Ez kisebb hálózatokban nem igazán releváns, viszont olyan esetekben, ahol akár több száz eszköz is csatlakozik egymáshoz, akadályozó tényező lehet. A megvalósított hálózatban ez a kis késleltetés elhanyagolható.

Az eszközök megfelelően működnek együtt, az új egységek beléptetése a hálózatba egyszerűen történt.

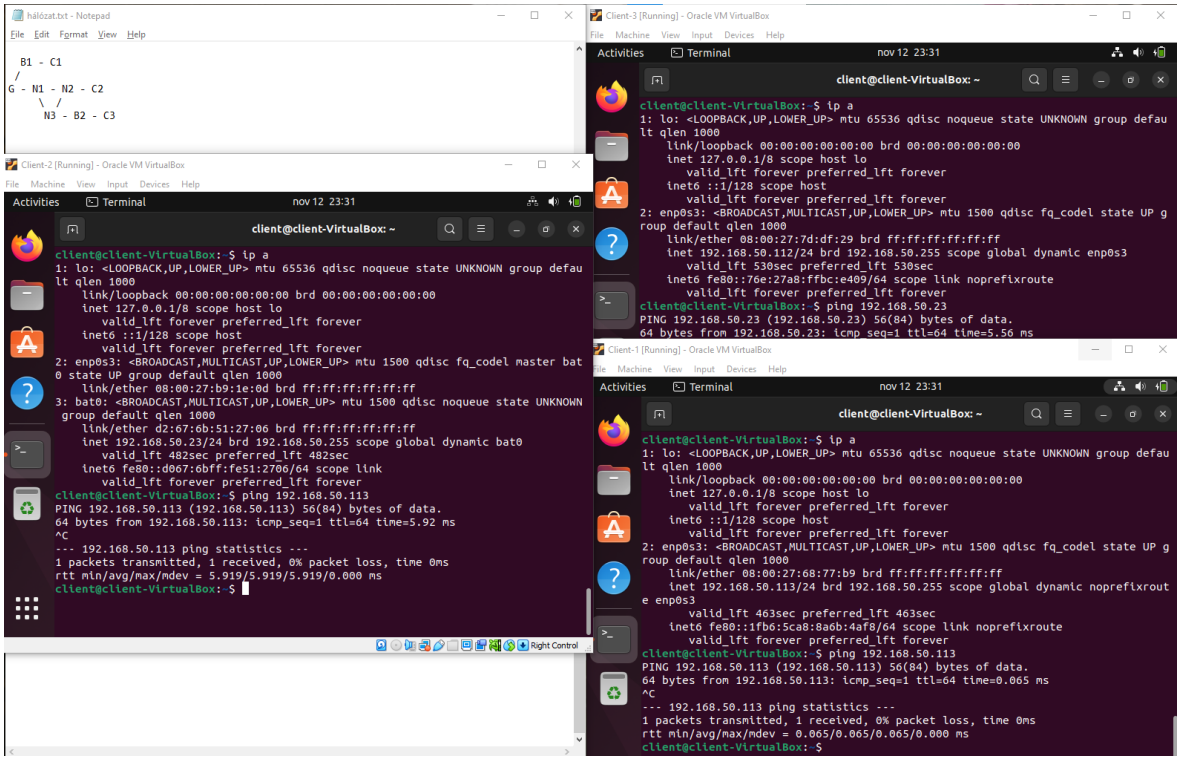
A rendszer telepítőjének GitHub-ról történő letöltése nagyban gyorsította a rendszer installációjának folyamatát, egyedül internetkapcsolatra volt szükség az egyes egységeken.

Batman-adv képességekkel nem rendelkező kliens eszközök nem észlelik, hogy valójában egy mesh hálózatot használnak. Hálózatba léptetésük a szokottól nem különbözik, IP címet is szolgáltat a rendszer számukra amennyiben ez az opció van kiválasztva. Eltérés a nem „bridge” eszközre csatlakozó batman-adv képességekkel rendelkező rendszereknél észlelhető, itt a terminál segítségével kell az interfészt hozzáadni a mesh hálózathoz. Csatlakozás után a kliens eszköz csatlakozási indikátora azt jelzi, hogy az interfész nem rendelkezik címmel. A jelenség azért jelentkezik, mert az

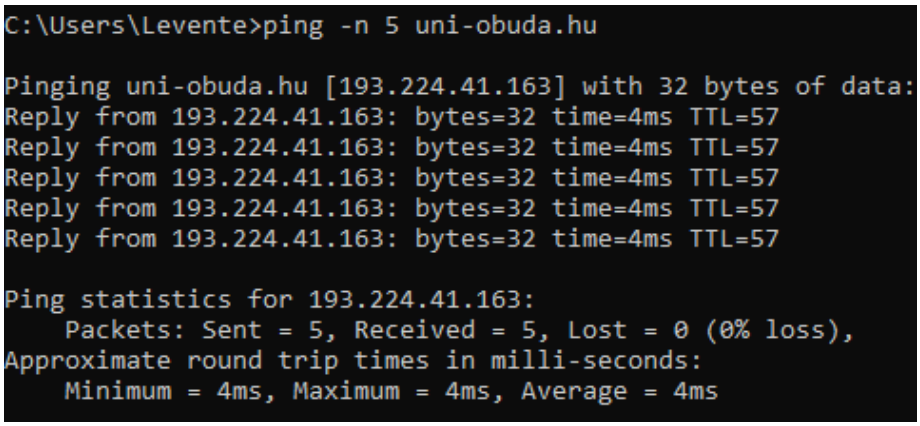


25. ábra: Kliens gépek hálózat elérésének tesztje.

interfészt mesh hálózatba léptetése után a virtuális batman-adv interfész kezeli, címet is ez a virtuális interfész fog kapni.



27. ábra: Kliensek el tudják érni egymást.



26. ábra: Virtuális gépeket futtató hoszt gép hálózati képessége.

9. Tesztesetek

A fejezetben a szakdolgozatban megvalósított rendszer pár tesztesete kerül bemutatásra röviden.

9.1 Tesztek csoportosítása

A teszteseteket céljuk szerint csoportosítottam.

Ezek a csoportok:

- Kisebb méretű funkcionalitást igazoló tesztek,
- Nagyobb méretű, a rendszer egészét vizsgáló tesztek.

9.2 Kisebb méretű funkcionalitást igazoló tesztek

Célja a rendszer tulajdonságainak elszigetelt vizsgálata. Általában 2, maximum 3 eszköz vett részt ilyen tesztesetekben.

9.2.1 Telepítés tesztelése

Szükséges követelmények:

1. A szkript csak jogosultsággal futtatható csak a szkript,
2. A rendszerhez szükséges csomagok települnek,
3. A szolgáltatások megfelelő állapotba kerülnek,
4. A fájlok megfelelő helyre másolódnak.

Tesztelés során a telepítő szkript funkcióit vizsgáltam. Futtatása után a követelményekben említett pontok alapján a rendszer vizsgálatával megbizonyosodhattunk, hogy a telepítő helyesen működik.

A tesztesetek megfelelően lefutottak, a követelmények teljesültek.

9.2.2 Batman-adv tesztek

A működés alapját biztosító rendszert vizsgáltam. Fő céljuk a helyes hálózati működés tesztelése.

Szükséges követelmények:

1. Sikeres telepítés,
2. Kommunikáció létrejövetele két eszköz között,
3. Több egymással közvetlen nem kapcsolódó eszköz között sikeres kommunikáció,
4. Hálózati hurkok kezelése.

Tesztelés során több eszközön telepíve lett a batman-adv rendszer, majd ezek között az eszközök között lettek megvizsgálva a hálózati funkciók.

A tesztek sikeresen lefutottak, a rendszer az elvártak szerint működik.

9.2.3 Eszközök kapcsolati tesztjei

A tesztelés során a különböző hálózati beállításokat vizsgáltam. Fő célja a kapcsolatok működésének tesztelése.

Szükséges követelmények:

1. Gateway csomópontok internetszolgáltatása a belső hálózatba,
2. Bridge olyan eszköz „mesh” hálózatba léptetése, amely nem rendelkezik batman-adv funkciókkal,
3. Eszközök egymát való elérése belső hálózatban.

Tesztelés során egy három eszközből álló teszthálózat lett kialakítva, egy gateway, egy bridge és egy kliens eszközzel.

A tesztesetek megfelelően lefutottak, a követelmények teljesültek.

9.2.4 Plugin teszt, DHCP plugin teszt

Fő célja a pluginkezelő, és a telepített plugin működésének vizsgálata.

Szükséges követelmények:

1. Pluginkezelő sikeres telepítése,
2. Plugin helyes installálása pluginkezelővel,
3. Plugin helyes installálása kézzel,
4. Telepített plugin helyes működése.

A tesztek során a telepítő helyes installációját és működését vizsgáltam. Telepítés után a pluginkezelőnek elérhetőnek kell lennie, illetve sikeresen tudnia kell pluginokat telepíteni. Plugin telepítése után helyes működést kell produkálnia, és a fájljainak megfelelő helyen kell elhelyezkednie.

A telepített DHCP a plugin tesztek alatt sikeresen települt, és szolgáltatott címeket.

A tesztesetek megfelelően lefutottak, a követelmények teljesültek.

9.2.5 Felhasználói felület tesztek

Fő célja a felhasználói felület funkcionalitásainak vizsgálata.

Szükséges követelmények:

1. A különböző oldalak elérhetőek,
2. Az egyes beállítások megfelelően működnek.

A tesztek során az oldalak elérhetősége lett kivizsgálva, illetve az oldalakon található beállítások. A Beállítások használata után a rendszer állapotának vizsgálata történt ezzel igazolva a helyes működést.

A tesztesetek megfelelően lefutottak, a követelmények teljesültek.

9.2.6 Wifi roaming teszt

A teszt során a csomópontok vezeték nélküli átcsatlakozási képessége kerül tesztelésre. Fő célja a csomópontok leszakadását megakadályozó képességének vizsgálata.

Szükséges követelmények:

1. A csomópont ténylegesen átcsatlakozik,
2. A legerősebb hálózatra csatlakozik,
3. Megfelelően authentikált hálózatra csatlakozik.

A tesztelés során a csomópont több ugyanolyan tulajdonságú hálózat közül tudott valamelyikre rácsatlakozni. Csatlakozás után a jelenlegi kapcsolatot a Hotspot megszüntetése bontotta, és így a csomópont muszáj volt újracsatlakozni.

A tesztesetek megfelelően lefutottak, a követelmények teljesültek.

9.3 Nagyobb méretű, a rendszer egészét vizsgáló tesztek

Célja a rendszer egészének tesztelése, megvalósított rendszerrészek együttműködésének vizsgálata. Tesztekben több, esetenként akár 8-10 eszközt használtam.

9.3.1 Nagyobb méretű hálózati teszt

A teszt során a 8fejezetben megvalósított hálózaton végeztem a teszteket.

Célja kisebb egységek együttműködésének vizsgálata.

Követelmények:

1. A hálózat tagjai elérik egymást,
2. A hálózat átviteli sebessége megfelelő,
3. Nem batman-adv képes eszközöknek is van lehetőségük csatlakozni a hálózathoz,
4. Internetelérés biztosítása hálózatbeli eszközök számára,
5. Az eszközök konfigurálhatóak hálózaton keresztül

A tesztek során a hálózat vizsgálatához használható parancsokkal és eszközökkel megbizonyosodhattunk a hálózati tagok egymást való elérési képességéről.

Hálózat átviteli sebességet virtuális klienseken mérve ≈36 Mbps (hálózaton kívüli virtuális gép átviteli sebessége: ≈31 Mbps) az alacsonyabb a virtuális gépek miatt, viszont tesztekből megállapítható, hogy nincs negatív hatással a hálózat az átvitelre.

A bridge eszközök tesztjei során megbizonyosodhattunk, hogy olyan eszköz is csatlakozni tud, amely nem rendelkezik batman-adv hálózati képeségekkel.

A gateway képes internetet biztosítani a hálózatnak. Hálózaton kívüli átviteli sebességek méréséhez elengedhetetlen az internetelérés.

Az eszközök a kliensek böngészőiből elérhetők, amennyiben a kliens a hálózat része.

A vizsgált tesztesetek megfelelően lefutottak, követelmények teljesültek.

10. További fejlesztési lehetőségek

10.1 Konfigurációs felület

Fejlesztésem során a konfigurációs felület kinézete nem volt prioritás, ezért az oldalak megjelenése elég egyszerű. Jövőben grafikai fejlesztések kerülhetnek sorra.

A csomópontok jelenleg csak mélyebb beállítási lehetőséget kínálnak, tehát külön-külön kell beállítani az egyes beállításokat. Ez nagyobb szabadságot eredményez, viszont hozzá nem értő felhasználók számára bonyolult lehet. Egy „könnyített" felhasználói felület létrehozása is lehetséges a jövőben, amely egy magasabb absztrakciós felületet biztosít a felhasználóknak.

Mivel Raspberry OS-re fejlesztettem a rendszer, ezért csak ennek a disztribúciónak tulajdonságaihoz illeszkedik, illetve ezen a platformon volt hibakeresés. Jövőben a telepítő felismerheti az eszközt, amelyen éppen fut, és a megfelelő szkripteket másolhatja a célmappákba. Ezt a rendszer széttagoltsága elősegíti, mivel csak olyan részeit kell a programnak cserélni, ahol disztribúcióbeli különbségek vannak.

10.2 Plugin rendszer

Egy biztonsági megvalósítás javításán célszerű a jövőben dolgozni. A pluginoknak célszerű lehet egy kulcs alapú ellenőrzés. Ilyen módon még ha a központi ellenőrzött plugintárolóban az egyik bejegyzést módosítják, a kulcsok alapján ezt a módosítást ki lehet szűrni.

10.3 Plugin kezelő

A plugin rendszernél említett kulcs ellenőrzés beépítésre kerülhet a pluginkezelőbe.

11. Összefoglalás

11.1 Magyar

A szakdolgozat elkészítése során megterveztem és megvalósítottam egy decentralizált hálózat kiépítéséhez alkalmas rendszert. A hálózat decentralizált tulajdonságát az nyújtja, hogy több egyenrangú eszközből építhető ki. Ezek az eszközök modulárisak, tehát a hálózat könnyen ellátható új tulajdonságokkal. A hálózat kialakításához a szakdolgozat során megvalósított eszközök a könnyű és gyors hálózat kiépítését célozzák meg, ezért megalkottam egy felhasználói felület is. Modularitásuk eléréséhez a megvalósítás során elkészítettem egy plugin rendszert. A pluginok egy megfelelő működéshez szükséges sémán kívül tetszőlegesen módosíthatóak, mind kinézet és funkcionalitás terén. A szakdolgozat során egy minta plugint is létrehoztam, amely használatával DHCP szolgáltatás biztosítható a hálózat számára. Megvalósítottam egy pluginokat kezelő segédprogram is, a pluginok könnyebb telepítése és kezelése érdekében. A pluginkezelő működése a főként Linux rendszerekben megtalálható csomagkezelő működéséhez hasonló, tehát a pluginokat képes egy központi tárolóból letölteni, telepíteni és installálni külső beavatkozás nélkül.

A rendszer tesztelését is elvégeztem, két fázisban. Az első fázisú tesztek alapfunkciók és működési folyamatok verifikálására szolgáltak. A második fázisban a rendszer egészét vizsgáltam, tehát itt már egy kiépített teszthálózaton folytak a tesztek. A tesztek sikeresen lefutottak, nagyobb hiba, illetve rendellenesség nem fordult elő a folyamatok során.

A projekt nyomon követéséhez és a könnyebb fejlesztés érdekében elkészítettem egy felépítési ábra és egy git tároló is.

11.2 English

During the thesis I designed and implemented a system suitable for building a decentralized network. The decentralized feature of the network is that it can be built from multiple peer devices. These devices are modular, so the network can be easily equipped with new features. The tools implemented in this thesis aim at building a network easily and quickly, and therefore I have also created a user interface. To achieve their modularity, I created a plugin system during the implementation. The plugins can be modified freely, both in terms of appearance and functionality, according to a schema for proper operation. During the thesis, I also created a sample plugin that can be used to provide DHCP service to the network. I have also implemented a plugins management utility to make plugins easier to install and manage. The functionality of the plugin manager is like the package manager found mainly in Linux systems, i.e. it can download, manage, and install plugins from a central repository without external intervention.

I also tested the system in two phases. The first phase tests were used to verify basic functionality and operational procedures. In the second phase, I tested the system, i.e.

tests were carried out on a deployed test network. The tests ran successfully, with no major errors or anomalies in the processes.

I also created a build diagram and a git repository to keep track of the project and for easier development.

12. Referenciák

[1]: Katrina Kwok, What is Wi-Fi Roaming and how it works, 2021, https://www.mercku.com/2021/07/28/what-is-wi-fi-roaming-and-how-it-works/,
Utoljára megtekintve: 2022.04.21

[2]: Laura Garcia, Jose M. Jimenez, Miran Taha,
Jaime Lloret, Wireless Technologies for IoT in Smart Cities, 2018,https://www.researchgate.net/publication/324460686_Wireless_Technologies_for_IoT_in_Smart_Cities,
Utoljára megtekintve: 2022.04.21

[3]: 802.11s based wireless mesh network, 2022, https://openwrt.org/docs/guide-user/network/wifi/mesh/80211s,
Utoljára megtekintve: 2022.04.21

[4]: B.A.T.M.A.N. General questions, https://www.open-mesh.org/projects/batman-adv/wiki/Faq,
Utoljára megtekintve: 2022.04.21

[5]: batman-adv, https://www.kernel.org/doc/html/v4.19/networking/batman-adv.html,
Utoljára megtekintve: 2022.04.21

[6]: Hostapd, https://wiki.gentoo.org/wiki/Hostapd,
Utoljára megtekintve: 2022.04.21

[7]: How Does WiFi Work? 2019, https://sookocheff.com/post/networking/how-does-wifi-work/,
Utoljára megtekintve: 2022.04.21

[8]: Internet speed, https://www.speedtest.net/
Utoljára megtekintve: 2022.04.21

[9]: Internet speed, https://fast.com/,
Utoljára megtekintve: 2022.04.21

[10]: ODROID-XU4, https://www.hardkernel.com/shop/odroid-xu4-special-price/,
Utoljára megtekintve: 2022.04.21

[11]: PainlessMesh Technical Documentation, 2018, https://gitlab.com/painlessMesh/painlessMesh/-/wikis/home,
Utoljára megtekintve: 2022.04.21

[12]: Raspberry Pi 4 Tech Specs, https://www.raspberrypi.com/products/raspberry-pi-4-model-b/specifications/,
Utoljára megtekintve: 2022.04.21

[13]: README.md, 2019, https://gitlab.com/painlessMesh/painlessMesh,
Utoljára megtekintve: 2022.04.21

[14]: TP-Link Archer C7 (2013) review,
https://routerchart.com/tp-link/tp-link-archer-c7-c7-173,
Utoljára megtekintve: 2022.04.21

[15]: TP-Link Archer C7 AC1750, 2021, https://openwrt.org/toh/tp-link/archer_c7,
Utoljára megtekintve: 2022.04.21

[16]: Marshall Brain & Talon Homer, How WiFi Works, 2021, https://computer.howstuffworks.com/wireless-network.htm,
Utoljára megtekintve: 2022.04.21

[17]: Yoppy, R Harry Arjadi, Endah Setyaningsih, Priyo Wibowo, M I Sudrajat, Performance Evaluation of ESP8266 Mesh Networks, 2019, https://iopscience.iop.org/article/10.1088/1742-6596/1230/1/012023/pdf,
Utoljára megtekintve: 2022.04.21

13. Melléklet

1 – Csomópont felépítése.png