УНИВЕРЗИТЕТ У НОВОМ САДУ
ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА У НОВОМ САДУ

Предраг Петковић

Очување корисничких података приликом миграције са Андроида на РДК Линукс

МАСТЕР РАД

Нови Сад, 2024.

УНИВЕРЗИТЕТ У НОВОМ САДУ ● **ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА**
21000 НОВИ САД, Трг Доситеја Обрадовића 6

КЉУЧНА ДОКУМЕНТАЦИЈСКА ИНФОРМАЦИЈА

Редни број, **РБР**:		
Идентификациони број, **ИБР**:		
Тип документације, **ТД**:		Монографска документација
Тип записа, **ТЗ**:		Текстуални штампани материјал
Врста рада, **ВР**:		Дипломски – мастер рад
Аутор, **АУ**:		**Предраг Петковић**
Ментор, **МН**:		**проф. др Илија Башичевић**
Наслов рада, **НР**:		**Очување корисничких података приликом миграције са Андроида на РДК Линукс**
Језик публикације, **ЈП**:		Српски / латиница
Језик извода, **ЈИ**:		Српски
Земља публиковања, **ЗП**:		Република Србија
Уже географско подручје, **УГП**:		Војводина
Година, **ГО**:		**2024**
Издавач, **ИЗ**:		Ауторски репринт
Место и адреса, **МА**:		Нови Сад; трг Доситеја Обрадовића 6
Физички опис рада, **ФО**: (поглавља/страна/ цитата/табела/слика/графика/прилога)		**7/37/8/2/14/0/0**
Научна област, **НО**:		Електротехника и рачунарство
Научна дисциплина, **НД**:		Рачунарска техника
Предметна одредница/Кључне речи, **ПО**:		**Миграција, *set-top box*, Андроид, РДК, бежична мрежа, очување корисничких података, мрежна конфигурација**
УДК		
Чува се, **ЧУ**:		У библиотеци Факултета техничких наука, Нови Сад
Важна напомена, **ВН**:		
Извод, **ИЗ**:		**Овај рад се бави проблематиком очувања корисничких података приликом миграције са Андроид оперативног система на РДК Линукс, са посебним освртом на бежичне мреже. Циљ рада је развијање решења које омогућава очување и пренос бежичних конфигурација корисника са Андроида на РДК, како би уређај могао аутоматски да се повеже на претходно сачуване мреже. У раду је имплементирана апликација која парсира XML датотеку са Андроида и пребацује бежичне мреже на РДК користећи C++ и BASH скрипте. Резултати тестирања потврђују успешност решења, а даља побољшања могу укључивати унапређење алгоритма за динамичко скенирање доступних мрежа.**
Датум прихватања теме, **ДП**:		
Датум одбране, **ДО**:		
Чланови комисије, **КО**:	Председник:	**<име председника комисије>**
	Члан:	**<име члана комисије>** — Потпис ментора
	Члан, ментор:	**<име ментора>**

UNIVERSITY OF NOVI SAD ● **FACULTY OF TECHNICAL SCIENCES**
21000 NOVI SAD, Trg Dositeja Obradovića 6

KEY WORDS DOCUMENTATION

Accession number, **ANO**:		
Identification number, **INO**:		
Document type, **DT**:		Monographic publication
Type of record, **TR**:		Textual printed material
Contents code, **CC**:		Master Thesis
Author, **AU**:		**Predrag Petković**
Mentor, **MN**:		**Ilija Bašičević, PhD**
Title, **TI**:		**Preservation of User Data During Migration from Android to RDK Linux**
Language of text, **LT**:		Serbian
Language of abstract, **LA**:		Serbian
Country of publication, **CP**:		Republic of Serbia
Locality of publication, **LP**:		Vojvodina
Publication year, **PY**:		**2024**
Publisher, **PB**:		Author's reprint
Publication place, **PP**:		Novi Sad, Dositeja Obradovica sq. 6
Physical description, **PD**: (chapters/pages/ref./tables/pictures/graphs/appendixes)		**7/37/8/2/14/0/0**
Scientific field, **SF**:		Electrical Engineering
Scientific discipline, **SD**:		Computer Engineering, Engineering of Computer Based Systems
Subject/Key words, **S/KW**:		**Migration, set-top box, Android, RDK, wireless network, user data preservation, Wi-Fi configuration**
UC		
Holding data, **HD**:		The Library of Faculty of Technical Sciences, Novi Sad, Serbia
Note, **N**:		
Abstract, **AB**:		**This thesis addresses the issue of preserving user data during the migration from the Android operating system to RDK Linux, with a specific focus on wireless networks. The aim of the thesis is to develop a solution that enables the preservation and transfer of user Wi-Fi configurations from Android to RDK, so that the device can automatically connect to previously saved networks. The thesis implements an application that parses the XML file from Android and transfers the Wi-Fi networks to RDK using C++ and Bash scripts. The test results confirm the success of the solution, with potential improvements including the enhancement of the algorithm for dynamic scanning of available networks.**
Accepted by the Scientific Board on, **ASB**:		
Defended on, **DE**:		
Defended Board, **DB**:	President:	**<ime predsednika komisije>**
	Member:	**<ime člana komisije>**
	Member, Mentor:	**<ime mentora>**

Menthor's sign

Захвалност

Захваљујем се ментору проф. др Илији Башичевићу, Јелени Живанић, као и осталим члановима тима на стручној помоћи и саветима током израде овог рада.

Захваљујем се члановима породице и пријатељима на неизмерној подршци током целокупног студирања.

САДРЖАЈ

Списак слика

Списак табела

Скраћенице

RDK	- *Reference Design Kit*
XML	- *eXtensible Markup Language*
HDR	- *High Dynamic Range*
SSID	- *Service Set Identifier*
OTA	- *Over-The-Air*
USB	- *Universal Serial Bus*
BIOP	- *Broadcast Inter-ORB Protocol*
API	- *Application Program Interface*
IPC	- *Inter-Process Communication*
BASH	- *Bourne Again Shell*

1. Увод

Брзи развој дигиталне телевизије и све већа потражња за што бољим корисничким искуством довели су до значајних напредака у технологији *set-top box* уређаја. Андроид, као широко прихваћена платформа, пружио је флексибилно и познато окружење за кориснике, омогућавајући им да уживају у великом броју апликација и услуга на својим кућним забавним системима. Међутим, појава РДК Линукса је увела нову парадигму у развоју *set-top box* уређаја нудећи прилагодљивију и ефикаснију платформу за пружаоце уређаја.

РДК Линукс, који је дизајниран специфично за повезане домове, пружа модуларну платформу отвореног кода, која унапређује контролу и флексибилност над функционалностима уређаја. Миграција са Андроида на РДК на *set-top box* уређајима доноси неколико предности, укључујући бољу интеграцију са позадинским (енг. *backend*) системима, побољшану сигурност и оптимизоване перформансе. Поред тога, природа отвореног кода РДК-а омогућава пружаоцима услуга да прилагоде корисничко искуство у складу са својим потребама, што потенцијално смањује трошкове и убрзава иновације.

Један од кључних аспеката ове миграције јесте осигурање беспрекорног преласка за кориснике, посебно у погледу задржавања њихових постојећих подешавања и преференција. Задатак овог рада јесте да сачува податке запамћених бежичних мрежа (*Wi-Fi* конфигурација), као и да омогући повезивање на одговарајућу мрежу када се РДК упали након миграције. Као решење је направљена извршна датотека која ће вршити миграцију бежичних мрежа. Она се позива као Линукс команда и прослеђује јој се XML датотека која садржи сачуване мреже са Андроида. Ова извршна датотека је направљена од једне C++ датотеке. РДК је Линукс систем заснован на *Yocto* систему изградње (енг. *build system)*.

Рад се састоји из седам поглавља.

У другом поглављу су наведене теоријске основе и алати који су коришћени приликом израде решења.

У трећем поглављу је описана софтверска архитектура решења, док је у четвртом поглављу описана програмска израда тог решења.

У петом поглављу су приказани резултати тестирања.

У шестом поглављу је изведен закључак, док је у седмом и последњем поглављу набројана коришћена литература и референце.

2. Теоријске основе

У овом поглављу су појашњене технологије које су у употреби за реализацију програмског решења.

2.1 Бежичне мреже

Бежичне мреже, познате и као *Wi-Fi* мреже (*Wireless Fidelity*), трансформисале су начин повезивања на интернет и међусобне комуникације. Концепт бежичне комуникације датира из раног 20. века, са пионирима попут Гуљелма Марконија, који је демонстрирао потенцијал радио таласа за пренос информација. Међутим, тек крајем 20. века је идеја коришћења радио таласа за повезивање уређаја на интернет постала стварност. Развој *Wi-Fi* технологије је почео 1990-их, првенствено вођен потребом да се обезбеди повезаност без ограничења физичких каблова у пословном окружењу. Институт инжењера електротехнике и електронике (IEEE) успоставио је 802.11 стандард 1997. године, који је поставио темеље модерне бежичне мреже. Током година, овај стандард се развијао, са значајним побољшањима у брзини, сигурности и ефикасности, што је довело до његове широке примене широм света.

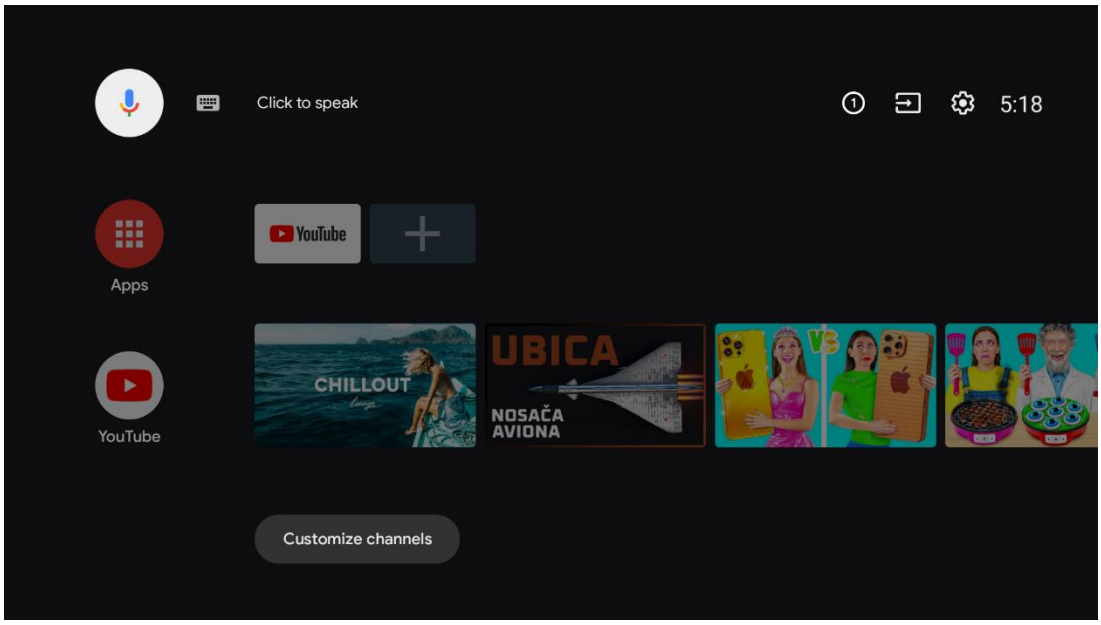
Употреба бежичних мрежа се драстично проширила од њиховог настанка, постајући саставни део приватног и професионалног живота. У домовима, бежичне мреже се користе за повезивање разних паметних уређаја, омогућавајући несметану комуникацију и аутоматизацију. У пословању, бежичне мреже обезбеђују основу за операције, подржавајући све, од комуникације запослених до услуга у облаку. Јавне бежичне мреже су такође постале уобичајене, нудећи повезивање на местима као што су кафићи, аеродроми и хотели. Како се *Wi-Fi* технологија наставља развијати, она игра кључну улогу у расту технологија следеће генерације комуникација.

2.2 Андроид

Андроид је платформа отвореног кода развијена од стране Гугла, дизајниран првенствено за мобилне уређаје као што су паметни телефони и таблети. Од свог изласка 2008. године, Андроид је постао једна од најпопуларнијих платформи на свету, захваљујући својој прилагодљивости, великој подршци за апликације, и широком екосистему уређаја. Андроид омогућава корисницима приступ милионима апликације, а његов модуларни дизајн омогућава произвођачима уређаја и инжењерима да прилагоде искуство коришћења према специфичним потребама.

2.2.1 Андроид ТВ

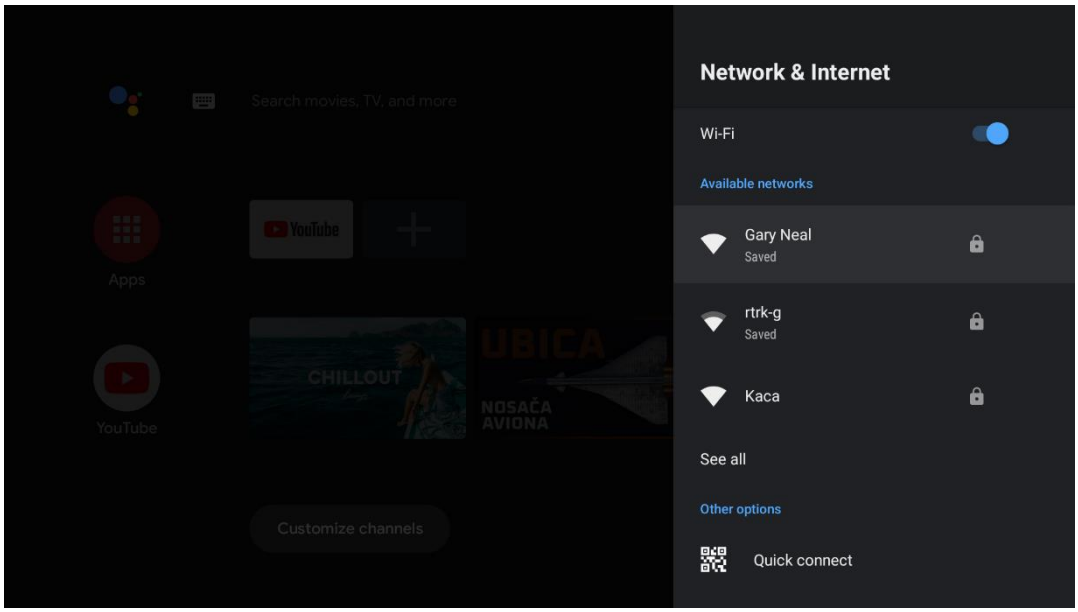
Андроид ТВ је специјализована верзија Андроида, оптимизована за коришћење на телевизорима и *set-top box* уређајима. Овај систем омогућава корисницима да претворе своје телевизоре у паметне уређаје са приступом широком спектру апликација, услога преноса уживо, и игара. Андроид ТВ за *set-top box* уређаје посебно је популаран код пружаоца услуга кабловске телевизије и интернет сервиса, јер омогућава пружање персонализованог и интерактивног садржаја корисницима. Корисничка спрега Андроид ТВ-а је прилагођена за употребу са даљинским управљачем, а такође подржава гласовне команде и интеграцију са паметним кућним уређајима. Са подршком за висококвалитетне видео формате, као што су 4K и HDR, као и за различите апликације, Андроид ТВ на *set-top box* уређајима пружа богато и динамично искуство гледања.



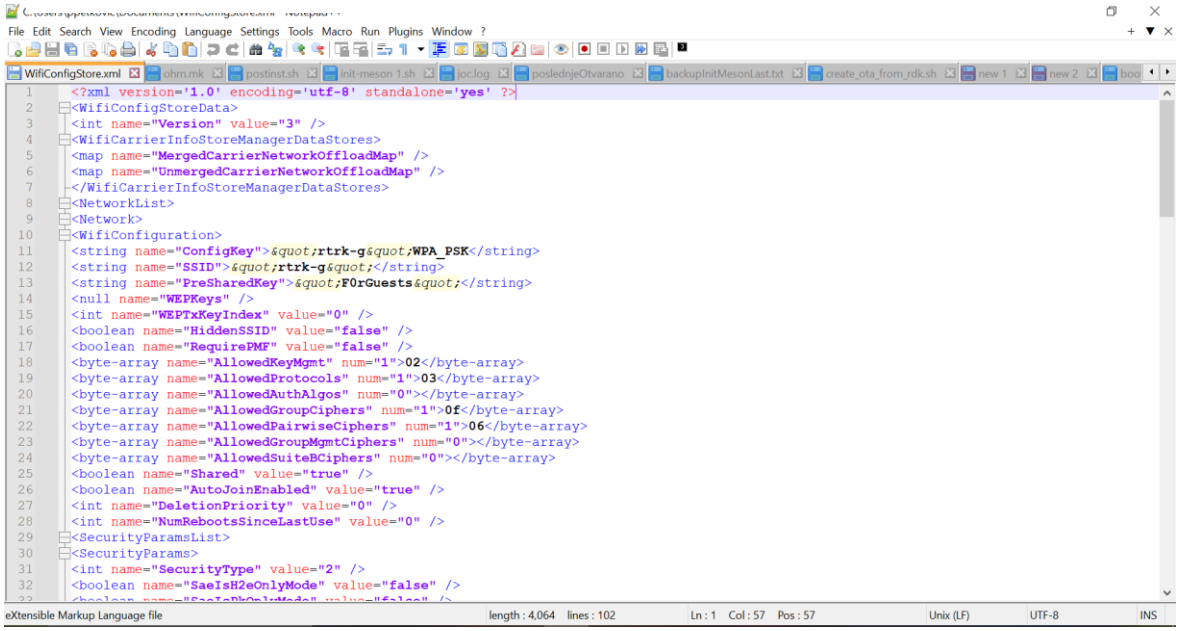
Слика 1 – Изглед *Amlogic set-top box* уређаја са Андроидом

2.2.2 Чување података о бежичним мрежама на Андроиду

Подаци о сачуваним бежичним мрежама се на Андроиду чувају у XML датотеци по имену *WifiConfigStore*. Ова датотека се на Андроид уређају налази на путањи */data/misc/apexdata/com.android.wifi*. Она чува све бежичне мреже на које је уређај икад био повезан или покушао да се повеже. Мрежа на коју је уређај тренутно повезан је означена параметром *ConnectChoice*. Датотека садржи чвор *NetworkList*, који садржи листу сачуваних мрежа. Свака мрежа је представљена чвором *Network*, као и чвором *WifiConfiguration*. У *WifiConfigStore* датотеци се налазе подаци о мрежи као што су име мреже (SSID), лозинка (*PreSharedKey*), начин заштите мреже, итд.



Слика 2 – Сачуване мреже на *Amlogic set-top box* уређају са Андроидом



Слика 3 – Садржај *WifiConfigStore.xml* датотеке

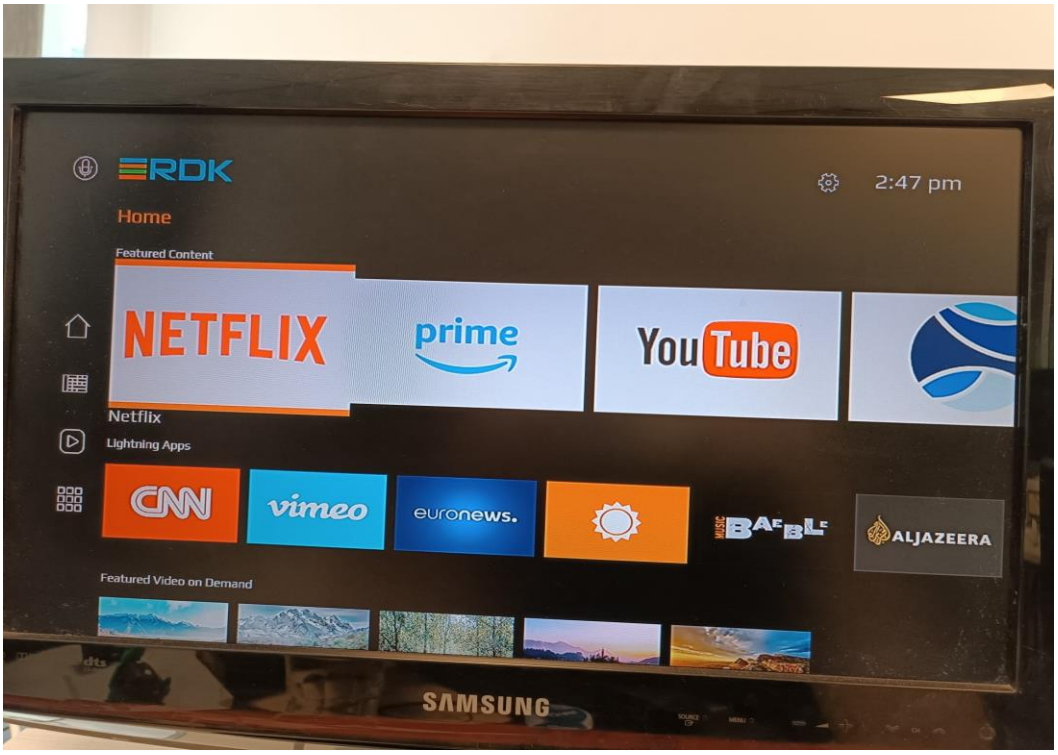
2.3 Линукс

Линукс је оперативни систем отвореног кода, који је познат по својој стабилности, сигурности и флексибилности. Првобитно развијен од стране Линуса Торвалдса 1991. године, Линукс је постао основа за широк спектар дистрибуција које се користе на серверима, рачунарима, мобилним уређајима, уграђеним системима, и супер рачунарима. Његова отвореност омогућава заједници програмера широм света да доприноси његовом развоју и прилагођава га различитим потребама.

2.3.1 РДК

РДК је софтверски оквир (енг. *software framework*) отвореног кода који се темељи на Линукс оперативном систему. РДК је дизајниран за убрзање развоја и имплементације апликација и услуга на *set-top box* уређајима, мрежним капијама (енг. *gateway*), и другим повезаним уређајима у домовима. Развијен од стране *RDK Management*, овај оквир пружа заједничку софтверску платформу која омогућава оператерима и произвођачима уређаја да креирају прилагођене корисничке спреге, апликације и функционалности специфичне за њихове мреже и услуге. Пошто се заснива на Линуксу, РДК користи предности стабилности, сигурности и флексибилности Линукса, омогућавајући ефикасно управљање ресурсима и висок ниво прилагодљивости.

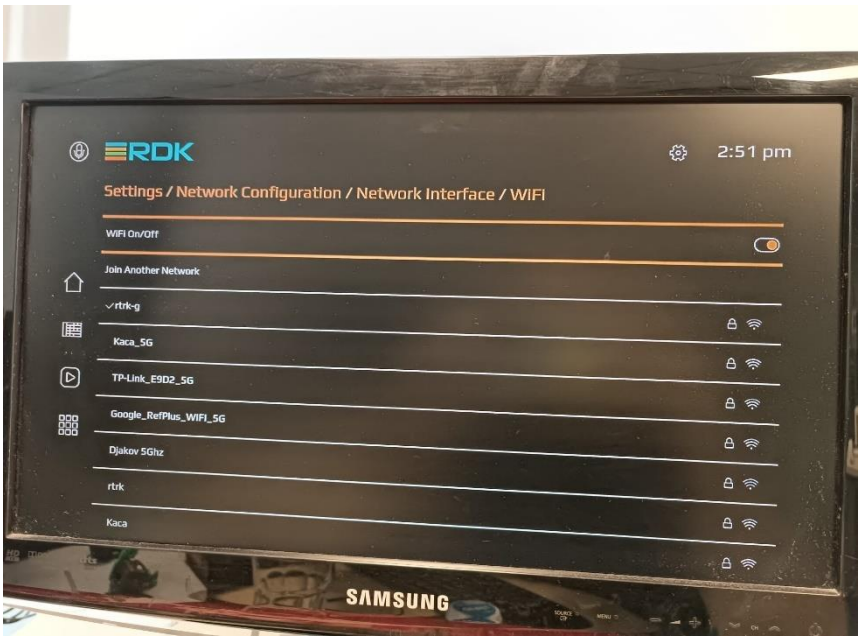
РДК подржава различите технологије, укључујући видео пренос уживо, гласовне команде, управљање мрежом и IoT интеграције, што га чини кључним алатом за испоруку напредних дигиталних услуга. Једна од главних предности РДК-а је његова модуларност, која омогућава оператерима да брзо реагују на промене на тржишту и прилагоде своје производе специфичним потребама корисника. Осим тога, РДК користи Линукс кернел за основне функционалности, што омогућава снажну подршку за мрежне протоколе, сигурност, и стабилност система. Због своје прилагодљивости и робусности, РДК је постао стандард кабловске и телекомуникационе индустрије, подржавајући милионе уређаја широм света.



Слика 4 – Изглед *Amlogic set-top box* уређаја са РДК-ом

2.3.2 Чување података о бежичним мрежама на РДК-у

Подаци о бежичним мрежама на РДК-у се чувају у конфигурационој датотеци по имену *wpa_supplicant.conf*. Ова датотека се налази на путањи */opt/secure/wifi* на РДК уређају. На РДК-у се у овој датотеци чувају подаци само о мрежи на коју је уређај био последње повезан, а не о свим мрежама као на Андроид уређајима. Ова датотека садржи податке о имену мреже, лозинци, као и начину заштите.



Слика 5 – Сачувана мрежа на *Amlogic set-top box* уређају са РДК-ом

Слика 6 – Садржај *wpa_supplicant.conf* датотеке

2.4 Yocto

Yocto је пројекат отвореног кода који пружа прилагодљив алат за креирање прилагођених Линукс дистрибуција за уграђене системе. Циљ *Yocto* пројекта је омогућити инжењерима и произвођачима да лако генеришу оперативне системе који су оптимизовани за специфичне хардверске платформе. *Yocto* не преставља готову Линукс дистрибуцију, већ сет алата и метаподатака који омогућавају корисницима да изграде свој сопствени систем према специфичним потребама. Помоћу *Yocto* алата, корисници могу прецизно дефинисати које пакете, функционалности и конфигурације желе укључити у свој систем, чиме се добија лаган, ефикасан и високо прилагодљив оперативни систем.

Yocto се користи у оквиру РДК пројекта како би се креирале прилагођене Линукс дистрибуције за *set-top box* уређаје и друге уграђене уређаје. РДК користи *Yocto* за аутоматизацију процеса изградње софтвера, омогућавајући оператерима и произвођачима уређаја да креирају специфичне верзије Линукс система које су оптимизоване за њихове призводе. Кроз *Yocto*, РДК омогућава прецизан одабир одговарајућих пакета, подешавање сигурносних параметара и оптимизацију перформанси. Овај приступ осигурава да сваки РДК уређај има тачно оне функционалности које су портебне за његово правилно функционисање, чинећи *Yocto* кључним алатом за испоруку ефикасних, стабилних и прилагодљивих решења у индустрији дигиталних уређаја.

Yocto користи систем рецепата (енг. *recipes*) и слојева (енг. *layers*) који дефинишу како ће се изградити одређени софтверски пакети, како ће се конфигурисати, и које ће компоненте бити укључене у коначни систем. *Yocto* компоненте:

- Рецепти: *Yocto* користи рецепте који дефинишу како се одређени софтверски пакет треба преузети, конфигурисати, превести (компајлирати), и инсталирати. Рецепти такође укључују информације о зависностима између пакета и потребним подешавањима. На пример, рецепт за кернел може

дефинисати верзију кернела, опције превођења, и закрпе (енг. *patch*) које треба применити.

- Слојеви: омогућавају модуларност и флексибилност. Слојеви садрже скуп рецепата и конфигурација специфичних за одређену архитектуру, апликације, или функционалности. На пример, један слој може бити специфичан за одређену архитектуру процесора, док други слој додаје софтверсје пакете за одређени уређај.
- *BitBake*: *Yocto* користи алат под називом *BitBake* који чита рецепте и управља процесом изградње. *BitBake* прати зависности између пакета и управља целокупним процесом превођења и паковања софтвера. Користећи конфигурационе датотеке и рецепте, *BitBake* осигурава да се сви потребни делови система преведу у одговарајућем редоследу. *BitBake* користи датотеке са наставком .bb које дефинишу како се одређени софтверски пакет преузима, конфигурише, преводи, и инсталира. Оне садрже упутства за *BitBake* алат о томе које кораке треба предузети да би се пакет изградио и укључио у коначну слику система. Користе се такође и датотеке са наставком .bbappend које проширују или модификују постојеће .bb датотеке. Користе се за примену измена или додавање функционалности без потребе за мењањем оригиналне .bb датотеке, што омогућава бољу модуларност и одржавање рецепата.
- Процес изградње: корисник покреће процес изградње дефинишући циљеве (енг. *targets*) и конфигурацију. *Yocto* затим користи рецепте да преузме изворни код софтверских пакета, преводи их, и генерише коначне бинарне датотеке.
- Изградња слике система (енг. *system image*): на крају процеса, *Yocto* генерише слику система која укључује кернел, коренски систем датотека и све потребне пакете. Та слика се затим може користити за покретање на циљаном уређају.
- *Devtool*: алат у оквиру *Yocto* пројекта који омогућава инжењерима да лакше развијају, тестирају и прилагођавају рецепте и пакете у *Yocto* окружењу. Помоћу њега, инжењери могу брзо извући постојећи рецепт, направити промене у изворном коду, поново превести и тестирати промене, све унутар *Yocto* окружења. Ово значајно убрзава процес развоје и омогућава ефикасније прилагођавање софтверских компоненти.

2.5 Партиције

Партиције су одељци на медијуму за складиштење који служе за организацију и раздвајање различитих делова оперативног система и података. На уређајима са Андроидом и РДК-ом, партиције играју кључну улогу у томе како се оперативни систем учитава, покреће, и како управља подацима.

На Андроид уређајима, партиције су јасно дефинисане како би подржале различите функције оперативног система. Неке од кључних партиција укључују:

- */boot*: садржи основне компоненте које су потребне за покретање система.
- */system*: садржи главни део оперативног система, укључујући Андроид оквир и системске апликације.
- */data*: садржи корисничке податке, апликације, и њихове податке. Ово је место где се складиште све корисничке инсталације и личне поставке, где се између осталог налазе и подихи о бежичним мрежама, укључујући и *WifiConfigStore.xml* датотеку.
- */recovery*: садржи алат за опоравак система који омогућава кориснику да врати уређај на фабричка подешавања између осталих функционалности.
- */cache*: користи се за привремено складиштење података и убрзавање одређених операција система.

На РДК уређајима, попут *set-top box* уређаја, партиције су прилагођене специфичним потребама тих уређаја, који су обично мање фокусирани на корисничке апликације и више на стабилан и ефикасан рад у области испоруке медијског садржаја. Неке кључне партиције су:

- */boot*: слично као и код Андроида, садржи основне компоненте за покретање система.
- */rootfs*: главна партиција која садржи све неопходне датотеке оперативног система, апликације специфичне за РДК, основне сервисе итд.
- */data*: служи за складиштење корисничких података и подешавања система. На РДК-у, ово је често мања партиција у поређењу са Андроидом јер је фокус на стабилности и перформансама.
- */opt*: посебна партиција или директоријум који се користи за складиштење додатног софтвера, апликација, или модула који нису део основног оперативног система. Ова партиција је дизајнирана да буде флексибилно место за инсталацију и складиштење додатних компоненти које могу бити потребне за специфичне функционалности уређаја, али нису интегрални део основне

слике система. Датотека *wpa_supplicant.conf,* која чува податке о бежичној мрежи на РДК, се налази у овој партицији.

2.6 OTA пакет

OTA пакет је софтверски пакет који се користи за ажурирање уграђеног софтвера, оперативног система или апликација на уређајима попут паметних телефона, таблета, *set-top box* уређаја и других уграђених система, путем интернета. Ови пакети омогућавају произвођачима да испоручују ажурирања директно корисницима без потребе да они физички повезују своје уређаје са рачунаром или користе спољне медије попут USB флешева. OTA пакети омогућавају једноставну и брзу дистрибуцију ажурирања. Овај пакет обично садржи бинарне датотеке, скрипте за инсталацију, и друге компоненте које су потребне за ажурирање софтвера на уређају. OTA пакети су често потписани дигиталним потписом како би се осигурала њихова аутентичност и интегритет. Уређај проверава потпис пре него што примени ажурирање, како би се осигурало да је пакет дошао од поузданог извора. Када се OTA пакет преузме, уређај обично прелази у режим ажурирања (који може укључивати и поновно покретање), примењује промене из пакета, и затим се поново покреће са новом верзијом софтвера.

2.7 Миграција

При миграцији са Андроида на РДК се креира OTA пакет који у себи садржи конфигурацију која ће рећи које РДК партиције треба да се ажурирају. Ова конфигурација може, а и не мора да садржи */data* партицију. Укључивањем */data* партиције се врши фабричко ресетовање уређаја. Већина произвођача при ажурирању нове верзије софтвера задржава корисничке податке као што су пријављени корисник, шифре, преференције и слично. Тако и ова миграција у својој суштини оставља */data* партицију такву каква јесте са свим сачуваним корисничким подацима. Она може да буде шифрована (софтверски или хардверски), али не мора да буде. У овом раду је фокус на */data* партицију када није шифрована.

2.8 Побољшано корисничко искуство

Миграција са Андроид система на РДК Линукс захтева посебно прилагођена решења која омогућавају очување кључних корисничких података, као што су бежичне мреже. Овај проблем се може посматрати као део ширег изазова оптимизације перформанси *set-top box* уређаја у окружењу дигиталних телевизијских система. На истом уређају, *Amlogic set-top box* уређају, претходно истраживање је укључивало рад на унапређењу кеширања BIOP

порука у оквиру Андроид ТВ преносног слоја (енг. *Broadcast Stack*), чиме су значајно побољшане перформансе у погледу брзине и поузданости преносе мултимедијалног садржаја. Ова истраживања су допринела разумевању начина на који се подаци на *set-top box* уређају могу ефикасније обрађивати и чувати, што је од посебног значаја и у процесу миграције података са Андроида на РДК. Тиме је постављена основа за унапређење метода које обезбеђују конзистентност података и поуздан рад уређаја током преласка између различитих оперативних система, што значајно унапређује искуство корисника.

3. Концепт решења

У овом поглављу ће бити представљена архитектура слоја додатог у *Yocto* пројекат, као и рецепата који су од значаја за решење овог рада.

3.1 meta-migration

Слој који обавља функционалност миграције се зове *meta-migration*. Састоји се из неколико рецепата који обављају различите функције, а они су груписани у *recipes-core, recipes-connectivity* и *recipes-security*. Неки од ових рецепата су *android-partitions-parser, and-rdk-dec, bootloader-control* итд. Они најбитнији за очување корисничких података о бежичним мрежама су *initrdscripts* и *wifi-migration*.

3.2 initrdscripts

initrdscripts је *Yocto* рецепт који између осталих датотека садржи скрипту *init-meson.sh*. Ова скрипта је задужена за покретање уређаја и она се поставља као *init* скрипта на циљном уређају. Слој *meta-migration* модификује ову скрипту како би је прилагодио миграцији са Андроида на РДК. Поред главних ствари везаних за миграцију, ова скрипта је задужена за пребацивање корисничких података са Андроид */data* партиције на */data* партиције РДК-а, првенствено *WifiConfigStore.xml* датотеке.

3.3 wifi-migration

wifi-migration је рецепт који је задужен да парсира податке добијене у *WifiConfigStore.xml* датотеци и повеже се на прву мрежу на коју је могуће повезивање. Рецепт се састоји из пет датотека: *wifi_migration.cpp, wifi-migration.sh, wifi-migration.service, CMakeLists.txt* и *wifi-migration.bb*. У наставку ће свака од ових датотека бити детаљно објашњена.

3.3.1 wifi_migration.cpp

wifi_migration.cpp је датотека са C++ изворним кодом која врши главну функционалност *wifi-migration* рецепта. Од ове датотеке се прави извршна бинарна датотека која се извршава на уређају. Као улазни параметар се прослеђује *WifiConfigStore.xml* датотека, која се парсира. Извлаче се подаци мреже као што су назив, лозинка и начин заштите, и све те мреже се сачувају. Потом се проверава да ли постоји мрежа означена параметром *ConnectChoice*, и ако постоји, та мрежа ће бити означена као мрежа на коју уређај треба прво да покуша да се повеже. Мрежа постаје означена овим параметром тек када се уређај други пут повеже на њу.

Након тога креће процес повезивања, уколико уређај није већ повезан на неку мрежу. Ако постоји мрежа означена параметром *ConnectChoice*, уређај ће покушати прво на њу да се повеже, ако не постоји, онда ће пробати на прву мрежу у листи. Након што се покрене повезивање на мрежу, изврши се чекање од две секунде, и онда се опет проверава да ли је уређај успешно повезан на мрежу. Ако јесте, процес повезивања ће престати, а уколико није, процес ће се наставити док се уређај не повеже или док се не дође до краја листе мрежа.

3.3.2 wifi-migration.sh

wifi-migration.sh јесте BASH скрипта која је задужена за позив *wifi-migration* извршне датотеке која је направљена од *wifi_migration.cpp* датотеке. Ова скрипта прво чека да се покрену потребни сервиси чији се API-ји користе у *wifi_migration.cpp*. Након што се сервиси покрену, покреће се и *wifi-migration*, али скрипта осигурава то да *wifi-migration* буде позван само једном, при првом покретању уређаја.

3.3.3 wifi-migration.service

Датотеке са наставком *.service* су конфигурационе датотеке које дефинишу како треба покренути, зауставити, и управљати сервисима на Линукс системима који користе *systemd* као систем за иницијализацију. *Systemd* је систем и услужни менаџер за Линукс који покреће системске сервисе приликом дизања система и управља њима док је систем активан.

wifi-migration.service представља системски сервис који је задужен за покретање *wifi-migration.sh* скрипте. Овај сервис се покреће након сервиса чији API-ји се користе у изради програмског решења.

3.3.4 CMakeLists.txt и wifi-migration.bb

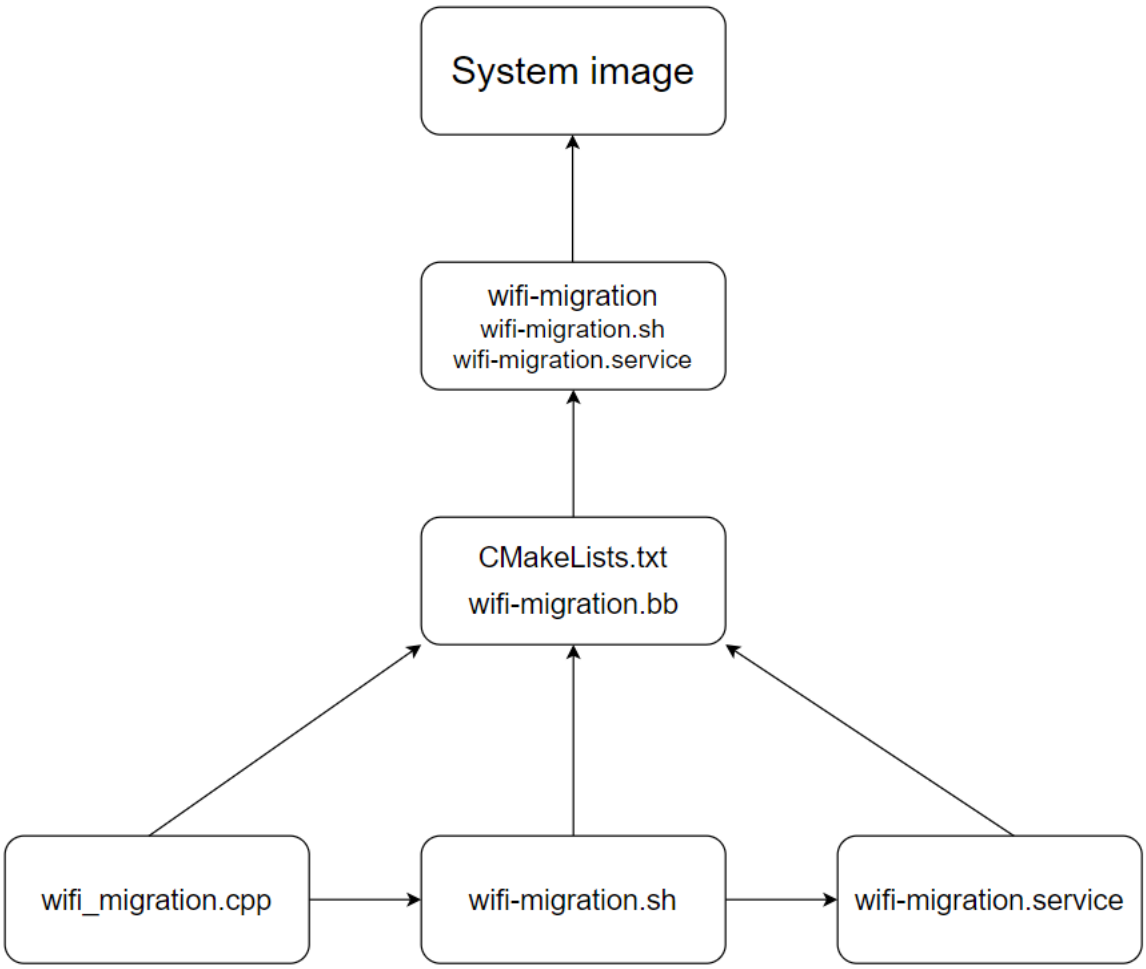
CMake је алат за аутоматизацију изградње софтвера који омогућава дефинисање процеса превођења и повезивања пројеката на преносив начин. Користи конфигурационе

датотеке назване *CMakeLists.txt* како би генерисао потребне *makefile*-ове или пројекте за различита развојна окружења.

CMakeLists.txt у овом решењу користи *CMake* за дефинисање конфигурације изградње за пројекат *wifi-migration*. Он спецификује минималну верзију *CMake*-а која је потребна, име пројекта, стандард C++ језика, изворне датотеке које треба превести, директоријуме које треба укључити, као и библиотеке са којима треба повезати пројекат. Такође, дефинише где ће резултујућа извршна датотека *wifi-migration* као и скрипта *wifi-migration.sh* бити инсталирани на циљном систему.

wifi-migration.bb је *BitBake* рецепт који *Yocto* користи за изградњу и паковање пројекта. Он дефинише метаподатке потребне за изградњу, као што су изворне датотеке које се укључују, зависности и алатни ланац који се користи. Такође спецификује да ће *CMake* бити коришћен као систем за изградњу. Поред тога, управља инсталацијом преведене извршне датотеке, скрипте, као и *systemd* сервисне датотеке на циљни систем. Рецепт се интегрише са *Yocto* процесом изградње, осигуравајући да пројекат буде правилно преведен, упакован и инсталиран у оквиру слике система.

CMakeLists.txt пружа упутства за изградњу пројекта, што се покреће кроз *wifi-migration.bb* рецепт. Рецепт користи *CMake* за конфигурацију и превођење пројекта, а затим пакује излаз (укључујући извршне датотеке, скрипту и сервисне датотеке) у финалну слику за дистрибуцију. У суштини, *CMakeLists.txt* управља специфичностима изградње, док *wifi-migration.bb* интегрише ову изградњу у *Yocto* окружење, управљајући зависностима, паковањем и инсталацијом.



Слика 7 – Архитектура *wifi-migration* рецепта

4. Програмско решење

У овом поглављу ће детаљније бити објашњене ствари наведене у претходном поглављу.

4.1 init-meson.sh

У овом делу ће фокус бити на томе како се извлаче подаци из */data* партиције Андроида и пребацују на РДК. Детаљи миграције неће бити објашњени, јер они не улазе у оквир овог рада, довољно је објаснити миграцију теоретски. Две најбитније функције за овај рад су *extract_android_data* и *restore_data.*

У функцији *extract_android_data* се прво прави директоријум */run/media/android_data* користећи команду *mkdir*. Овај директоријум служи као тачка монтирања (енг. *mount point*) за директоријум */dev/disk/by-partlabel/userdata* који представља */data* партицију Андроида. Потом се прави */run/media/migration_tmp_data* директоријум, на који се монтира привремени систем датотека (*tmpfs*). Након тога се *WifiConfigStore.xml* датотека копира са путање */run/media/android_data/misc/apexdata/com.android.wifi* на */run/media/migration_tmp_data*. Затим се сви монтирани директоријуми демонтирају.

У функцији *restore_data* се прво проверава да ли постоји датотека *WifiConfigStore.xml* на путањи */run/media/migration_tmp_data*. Ако постоји, прави се датотека */rootfs/data/android_wifi_data*, на коју се копира датотека *WifiConfigStore.xml*, она ће се налазити на овој путањи кад се РДК покрене.

Функција *extract_android_data* се позива током процеса монтирања и провере партиција, пре него што се изврши форматирање и инсталација новог система или покретање коренског система датотека. Функција *restore_data* се позива након што је коренски систем датотека монтиран и систем је спреман за покретање.

4.2 wifi-migration.cpp

Ова датотека садржи једну глобалну променљиву, а то је *globalConnectChoice*, која служи за смештање имена бежичне мреже која је означена као *ConnectChoice* у *WifiConfigStore.xml*.

Поље	**Опис**
string ssid	Име бежичне мреже
string preSharedKey	Лозинка бежичне мреже
string securityMode	Начин заштите бежичне мреже
string connectChoice	Служи за смештање имена бежичне мреже ако је означена *ConnectChoice* параметром, пре него што се име мреже обради за смештање u *globalConnectChoice*

Табела 1 – Поља структуре *NetworkConfig*

Функција	Опис
string getElementAttribute (xmlNode node, const string& attributeName)*	Враћа задати атрибут прослеђеног чвора *WifiConfigStore.xml* датотеке
string findChildTextByNameAttribute(xmlNode parent, const string& targetName)*	Претражује чворове децу прослеђеног родитељског чвора са *name* атрибутом. Потом проверава да ли се текстуални садржај чвора подудара са *targetName*, и ако се подудара враћа тај садржај
pair<string, string> parseConfigKey(const string& configKey)	Парсира *ConfigKey* чвор који садржи назив и начин заштите мреже, и враћа ове две вредности
string removeQuotes(const string& str)	Ако су први и последњи карактер стринга наводници, уклања их
optional<NetworkConfig> getGlobalConnectConfig(vector<NetworkConfig> networkConfigs)	Пролази кроз све бежичне мреже и враћа податке оне која је означена као *ConnectChoice*
int getCurrentWifiState()	Враћа статус повезаности на бежичну мрежу
void connect(string ssid, string passphrase, string securityMode, bool isGlobalConnectChoice)	Извршава повезивање на бежичну мрежу с прослеђеним подацима
bool connectToWifiNetwork(const NetworkConfig& config, bool isGlobalConnectChoice)	Ако уређај није већ повезан на мрежу, позива *connect* и потом извршава чекање од две секунде. Враћа да ли је уређај повезан на мрежу или не
void handleWifiNetworksConnection(const vector<NetworkConfig>& configs)	Извршава логику редоследа повезивања на мреже

Табела 2 – Функције у *wifi_migration.cpp* датотеци

wifi_migration.cpp користи *libxml2* API за руковање XML датотекама, а *NetworkManager (netsrvmgr)* и *IARMBus* користи за руковање бежичним мрежама.

libxml2 је библиотека отвореног кода која се користи за парсирање и манипулацију XML датотекама. Омогућава апликацијама да лако читају, пишу и модификују XML документе.

IARMBus је оквир за комуникацију између процеса (IPC) који се користи на РДК платформама. Омогућава различитим процесима да међусобно комуницирају и размењују податке. Такође омогућава различитим компонентама у РДК окружењу да ефикасно комуницирају, што је кључно за комплексне системе где различите апликације и сервиси морају да раде заједно како би обезбедили функционалност, као што је управљање мрежама.

NetworkManager је системски сервис на Линуксу који управља мрежним конекцијама, укључујући и бежичне мреже.

Програмско решење користи *IARMBus* за позивање функција које управљају бежичним мрежама, као што су провера тренутног стања мреже или повезивања на мрежу. *IARMBus* спрега служи као мост између апликације и система, и преко њега се заправо комуницира са *NetworkManager*-ом, који обавља стварне мрежне операције. Иако *NetworkManager* није директно референциран у коду, он је критичан за функционисање система, јер преко њега уређај остварује и одржава мрежне конекције.

Када се покрене *wifi_migration.cpp,* прво се изврши провера да ли је правилно прослеђен аргумент командне линије, који треба да буде *WifiConfigStore.xml* датотека.

Потом се позивају функције *IARM_Bus_Init* и *IARM_Bus_Connect* које извршавају иницијализацију и успостављају везу са *IARMBus*-ом.

Након тога се учитава *WifiConfigStore.xml* користећи функцију *xmlReadFile* библиотеке *libxml2*. Потом се тражи почетни чвор датотеке, користећи функцију *xmlDocGetRootElement,* чије име мора бити *WifiConfigStoreData*, што се проверава користећи функцију *xmlStrcmp* библиотеке *libxml2*. Након тога се међу децом почетног чвора тражи чвор по имену *NetworkList,* такође користећи *xmlStrcmp* функцију библиотеке *libxml2*. Потом се прави листа структура *NetworkConfig*, која ће заправо представљати листу пронађених бежичних мрежа. Затим се траже чворови по имену *Network* и *WifiConfiguration*, такође користећи *xmlStrcmp.* Потом се тражи параметар са именом *ConfigKey,* користећи функцију *findChildTextByNameAttribute*. Ова функција користи *xmlNodeGetContent* да узме садржај чвора. Овај чвор се потом прослеђује функцији *parseConfigKey*, одакле се добијају име мреже и начин заштите мреже, који се смештају у *NetworkConfig* структуру. Потом се тражи лозинка мреже, користећи *findChildTextByNameAttribute* за параметар са именом *PreSharedKey*. Ова лозинка се прослеђује функцију *removeQuotes*, да би се добила лозинка без наводника. И ова лозинка се смешта у *NetworkConfig* структуру. Потом се тражи чвор *NetworkStatus* који у себи има параметар *ConnectChoice*, чија се вредност такође смешта у *NetworkConfig* структуру. Затим, ако назив и лозинка нису празни, тренутна мрежа се смешта у листу структура *NetworkConfig*.

Након овог се пролази кроз све пронађене мреже и тражи ако нека мрежа садржи вредност *ConnectChoice* која није празна. Ако се пронађе оваква мрежа, она се прослеђује функцији *removeQuotes,* и таква вредност се смешта у глобалну променљиву *globalConnectChoice.*

Потом се позива функција *handleWifiNetworksConnection,* која прво дефинише променљиву *isConnected* која ће представљати стање повезаности на мрежу. Њена иницијална вредност је *false*. Потом се проверава да ли је *globalConnectChoice* празан, и ако није, позива се функција *getGlobalConnectConfig*, која ће вратити *NetworkConfig* структуру,

која ће се проследити функцији *connectToWifiNetwork*. Променљива *isConnected* прима повратну вредност функције *connectToWifiNetwork*. Након овог се проверава вредност променљиве *isConnected*, и ако је и даље *false*, пролази се кроз све остале мреже и позива се *connectToWifiNetwork*, све док *isConnected* не буде на *true*, или док се не дође до краја листе.

Функција *connectToWifiNetwork* прво проверава да ли је уређај повезан на неку бежичну мрежу користећи *getCurrentWifiState* функцију, и ако уређај јесте повезан, излази се из функције. Ако није повезан, позива се функција *connect*, и након ње се позива чекање, тј. спавање од две секунде. Ово се ради како би се уређају дало времена да се повеже на мрежу. На крају функције се проверава да ли се уређај успешно повезао користећи *getCurrentWifiState*, поредећи њену повратну вредност са макроом WIFI_CONNECTED. Функција *connectToWifiNetwork* враћа резултат овог поређења.

Функција *getCurrentWifiState* позива функцију *IARM_Bus_Call*, којој се каже да комуницира са *NetworkManager*-ом тако што јој се као један од параметара проследи IARM_BUS_NM_SRV_MGR_NAME, док је још један од параметара задужен да каже која ће се функционалност извршити или која ће се метода позвати. Тај параметар овде јесте IARM_BUS_WIFI_MGR_API_getCurrentState. Ова функција враћа *wifiStatus* који је набројивог типа *WiFiStatusCode_t* и део је *NetworkManager*-а. Његова вредност може бити WIFI_UNINSTALLED, WIFI_DISABLED, WIFI_DISCONNECTED, WIFI_PAIRING, WIFI_CONNECTING, WIFI_CONNECTED, WIFI_FAILED.

Функција *connect* прво параметром *isGlobalConnectChoice* проверава да ли је мрежа означена као *ConnectChoice*. Ако јесте, биће исписано да јесте *globalConnectChoice*, а ако није, биће исписано да није. Потом се проверава начин заштите. Досад је одрађено повезивање само на мреже чији је начин заштите WPA_PSK. Ако је мрежа заштићена на овај начин, *securityMode* ће попримити вредност NET_WIFI_SECURITY_WPA_PSK_AES, иначе ће бити NET_WIFI_SECURITY_NONE. Након овог се позива *IARM_Bus_Call*, којем се прослеђује IARM_BUS_WIFI_MGR_API_connect са називом, лозинком и начином заштите мреже који су заправо параметри ове функције.

4.3 wifi-migration.sh

У овој скрипти се налази функција *is_file_ready*, која проверава да ли у датотеци *netsrvmgr.log*, која се на уређају налази на путањи */var/lib/logs*, постоји испис *wifi_init() done*. Ова функција се позива у петљи све док се не пронађе овај испис, што означава то да је *NetworkManager* иницијализован. Након тога се проверава да ли на уређају постоји *WifiConfigStore.xml* датотека, и ако постоји, позива се *wifi-migration* извршна датотека генерисана од *wifi_migration.cpp* датотеке. Након тога се брише *WifiConfigStore.xml* како се

wifi-migration не би позивао сваки пут када се покрене уређај, већ само при првом покретању.

4.4 wifi-migration.service

Овај системски сервис зависи од *NetworkManager* сервиса, и покреће се након њега. *NetworkManager* сервис зависи од *IARMBus* сервиса и покреће се након њега. *wifi-migration* сервис извршава две команде. Прва је та да омогући извршавање *wifi-migration.sh* скрипте, а онда покреће ту скрипту, и њен излаз смешта у *wifi-migration.log* датотеку на путањи */opt/logs* на уређају.

4.5 CMakeLists.txt i wifi-migration.bb

CMakeLists.txt прави извршну датотеку *wifi-migration* од *wifi_migration.cpp*. Потом укључује *libxml2* директоријум са *header* датотекама. Затим повезује извршне датотеке *xml2* и *IARMBus* са *wifi-migration* извршном датотеком. Након тога инсталира *wifi-migration* и *wifi-migration.sh* на */usr/sbin* путању на уређају.

У *wifi-migration.bb* се наводе изворне датотеке које су потребне за изградњу рецепта, а то су *NOTICE* датотека, која садржи коришћену лиценцу, *CMakeLists.txt*, *wifi_migration.cpp*, *wifi-migration.service* и *wifi-migration.sh*. Ове датотеке се смештају у променљиву SRC_URI. Потом се наводе пакети од којих зависи изградња рецепта, а то су *libxml2* и *netsrvmgr*. Ове зависности се наводе у променљивој DEPENDS. Затим се дефинишу датотеке које ће бити паковане као део бинарног пакета. Извршна датотека *wifi-migration* и скрипта *wifi-migration.sh* се пакују у */usr/sbin* директоријум, док се *wifi-migration.service* пакује у *systemd* директоријум. Ово се дефинише у променљивој FILES_${PN}, где PN представља име пакета који се гради, што је у овом случају *wifi-migration*. Потом се позива *do_install_append* функција која је *BitBake* функција и она додаје додатне кораке инсталације након главне инсталације *do_install*. У овој функцији се сервис *wifi-migration.service* инсталира на предвиђену путању на уређају. Затим се променљивој SYSTEMD_SERVICE_${PN} додељује *wifi-migration.service*. Ова променљива дефинише сервисну датотеку повезану са пакетом. Ово омогућава то да *systemd* покрене *wifi-migration* сервис при покретању уређаја, да не мора ручно да се покреће.

5. Резултати

У овом поглављу ће бити приказани резултати тестирања и испитивања програмског решења, као и различити случајеви који су испитани. Под овим различитим случајевима се подразумева различит садржај *WifiConfigStore.xml* датотеке. Испитан је случај када се у датотеци налази само једна мрежа, као и случај када бежична мрежа која је доступна буде прва ла листи бежичних мрежа. Такође су испитани случајеви када мрежа која је доступна није прва на листи, већ су направљени тестови кад је друга и кад је трећа.

Досад описани случајеви нису садржали мрежу означену параметром *ConnectChoice*, али су такође испитани и случајеви када таква мрежа постоји, и када је доступна и када није.

```
root@AmlogicFirebolt:~# cat /opt/logs/wifi-migration.log
Waiting for netsrvmgr to initialize...
Waiting for netsrvmgr to initialize...
Waiting for netsrvmgr to initialize...
Waiting for netsrvmgr to initialize...
Waiting for netsrvmgr to initialize...
Waiting for netsrvmgr to initialize...
Waiting for netsrvmgr to initialize...
Waiting for netsrvmgr to initialize...
Waiting for netsrvmgr to initialize...
netsrvmgr initialized
IARM_Init group name = com.comcast.rdk.iarm.bus member name = wifi-migration
setting init done
Registering wifi-migration
Not yet connected to a Wi-Fi network...
Trying to connect to rtrk-g
rtrk-g is not global connect choice.
Connected to rtrk-g
Successfully connected to a Wi-Fi network.
```

Слика 8 – Уређај се повезује на прву мрежу са листе

```
root@AmlogicFirebolt:~# cat /opt/logs/wifi-migration.log
Waiting for netsrvmgr to initialize...
Waiting for netsrvmgr to initialize...
Waiting for netsrvmgr to initialize...
Waiting for netsrvmgr to initialize...
Waiting for netsrvmgr to initialize...
Waiting for netsrvmgr to initialize...
Waiting for netsrvmgr to initialize...
Waiting for netsrvmgr to initialize...
Waiting for netsrvmgr to initialize...
Waiting for netsrvmgr to initialize...
netsrvmgr initialized
IARM_Init group name = com.comcast.rdk.iarm.bus member name = wifi-migration
setting init done
Registering wifi-migration
Not yet connected to a Wi-Fi network...
Trying to connect to Gary Neal
Gary Neal is not global connect choice.
Not yet connected to a Wi-Fi network...
Trying to connect to rtrk-g
rtrk-g is not global connect choice.
Connected to rtrk-g
Successfully connected to a Wi-Fi network.
```

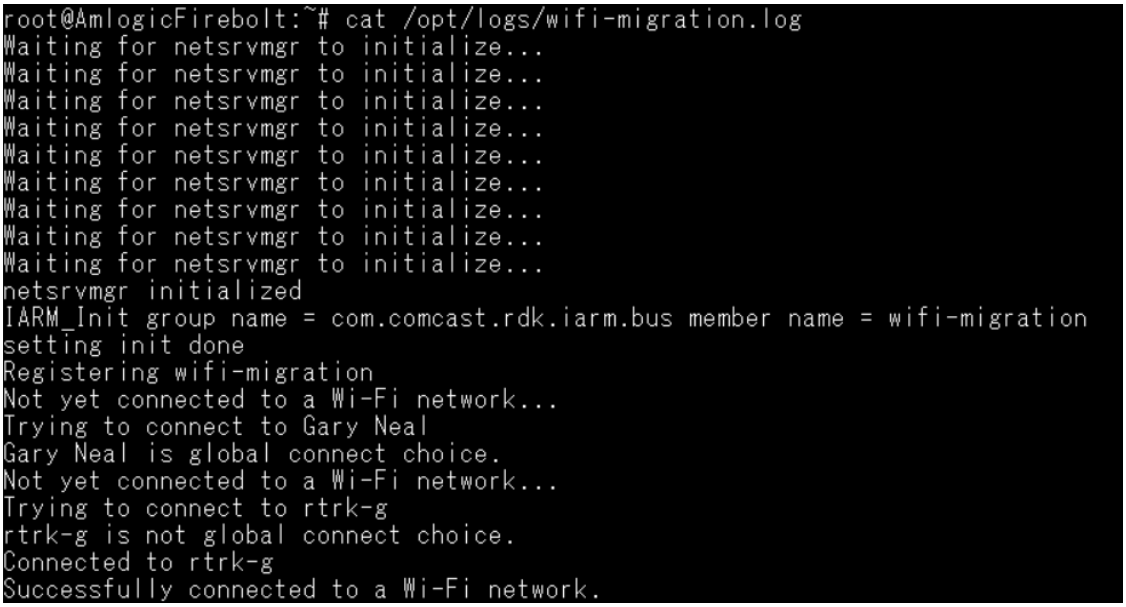
Слика 9 – Уређај се повезује на другу мрежу са листе

```
netsrvmgr initialized
IARM_Init group name = com.comcast.rdk.iarm.bus member name = wifi-migration
setting init done
Registering wifi-migration
Not yet connected to a Wi-Fi network...
Trying to connect to Gary Neal
Gary Neal is not global connect choice.
Not yet connected to a Wi-Fi network...
Trying to connect to Mihailo
Mihailo is not global connect choice.
Not yet connected to a Wi-Fi network...
Trying to connect to rtrk-g
rtrk-g is not global connect choice.
Connected to rtrk-g
Successfully connected to a Wi-Fi network.
```

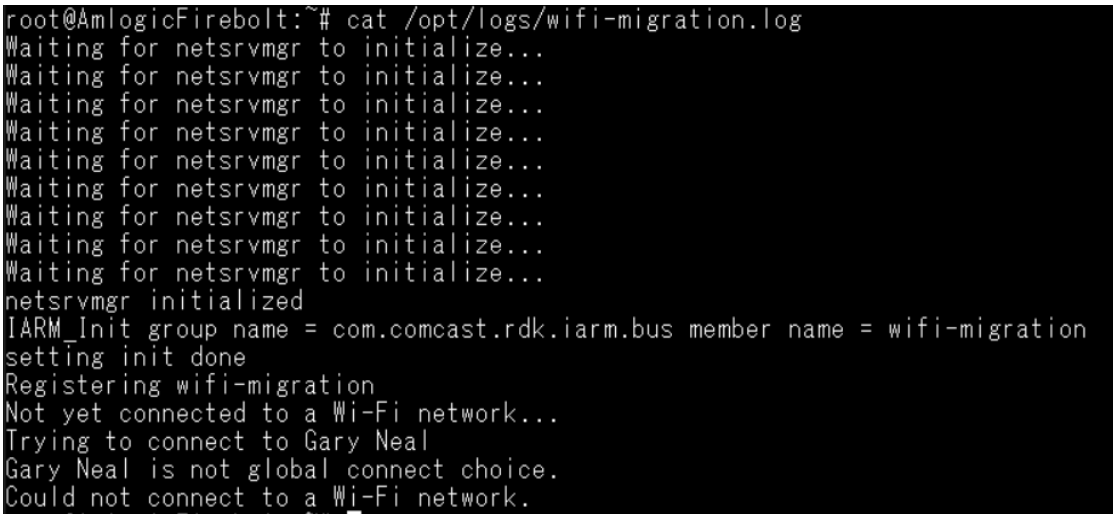
Слика 10 – Уређај се повезује на трећу мрежу са листе

```
root@AmlogicFirebolt:~# cat /opt/logs/wifi-migration.log
Waiting for netsrvmgr to initialize...
Waiting for netsrvmgr to initialize...
Waiting for netsrvmgr to initialize...
Waiting for netsrvmgr to initialize...
Waiting for netsrvmgr to initialize...
Waiting for netsrvmgr to initialize...
Waiting for netsrvmgr to initialize...
Waiting for netsrvmgr to initialize...
Waiting for netsrvmgr to initialize...
netsrvmgr initialized
IARM_Init group name = com.comcast.rdk.iarm.bus member name = wifi-migration
setting init done
Registering wifi-migration
Not yet connected to a Wi-Fi network...
Trying to connect to Gary Neal
Gary Neal is global connect choice.
Connected to Gary Neal
Successfully connected to a Wi-Fi network.
```

Слика 11 – Уређај успешно повезан на мрежу означену параметром *ConnectChoice*



Слика 12 – Мрежа означена параметром *ConnectChoice* није доступна



Слика 13 – Ниједна мрежа са листе није доступна

На сликама су представљени различити случајеви тестирања, где су приказани исписи за сваки покривени случај. Први случај је када се доступна мрежа налази на првом месту у *WifiConfigStore.xml* датотеци, док се код другог и трећег случаја доступна мрежа налази на другом и трећем месту. Четврти случај приказује то када је мрежа која је означена параметром *ConnectChoice* доступна, а пети случај кад није. Уређај ће увек покушати да се повеже на мрежу означену овим параметром, уколико она постоји. Шести случај покрива то када ниједна мрежа са листе није доступна.

Испитивањем се дошло до закључка да је чекање од две секунде након покушаја повезивања потребно и довољно како би алгоритам правилно функционисао. Извршено је

и тестирање када је чекање износило једну секунду, али тада апликације није имала очекивано понашање, јер једна секунда није довољна како би се уређај повезао на мрежу.

Сва испитивања су извршена на *Amlogic set-top box* уређају.

Слика 14 – *Amlogic set-top box* уређај

6. Закључак

У овом раду је детаљно обрађен процес миграције корисничких података са Андроида на РДК Линукс платформу, са посебним фокусом на очување и пренос конфигурација бежичних мрежа. Миграциони процес омогућава преузимање релевантних података са Андроид уређаја и њихово успешно интегрисање на РДК, чиме се обезбеђује несметано повезивање на бежичне мреже у оквиру новог окружења.

Кроз имплементацију овог решења, показано је да РДК омогућава високу флексибилност и модуларност, што је кључно за очување беспрекорног корисничког искуства. Успешно повезивање на мрежу након миграције на РДК показује да кључни кориснички подаци могу бити сачувани и пренети без губитка функционалности.

Даље унапређење решења би могло бити скенирање првенствено доступних мрежа након што се РДК упали, и да се након тога све мреже које се налазе у *WifiConfigStore.xml* датотеци, а нису међу доступним мрежама, избаце из листе структура *NetworkConfig*.

7. Литература

[1] „A Brief History of WiFi“, 2022. [Online]. Доступно: https://www.compareinternet.com/blog/a-brief-history-of-wifi/. [Посећено: 21. 8. 2024.]

[2] “Android - Overview”, 2023. [Online]. Доступно: Android - Overview (tutorialspoint.com). [Посећено: 21.8.2024.]

[3] “What is Android TV? Google’s smart TV platform fully explained”, 2023. [Online] Доступно: https://www.digitaltrends.com/home-theater/what-is-android-tv/. [Посећено: 21.8.2024.]

[4] “RDK Documentation”, 2022. [Online]. Доступно: https://wiki.rdkcentral.com/display/RDK/RDK+Documentation. [Посећено: 22.8.2024.]

[5] “Yocto Project Technical Overview” [Online]. Доступно: https://www.yoctoproject.org/development/technical-overview/. [Посећено: 23.8.2024.]

[6] “How Yocto Integrated in RDK” [Online]. Доступно: https://wiki.rdkcentral.com/display/RDK/How+Yocto+integrated+in+RDK. [Посећено: 23.8.2024.]

[7] “OTA Updates: What Are Over-The-Air Updates?”, 2024. [Online]. Доступно: https://www.esper.io/blog/the-evolution-of-android-ota-updates. [Посећено: 24.8.2024.]

[8] P. Petković, T. Valeev and I. Bašičević, "Enhancing Caching Efficiency of DSM-CC Data Carousel BIOP Messages for Android TV Broadcast Stack Virtual File System," *2024 Zooming Innovation in Consumer Technologies Conference (ZINC)*, Novi Sad, Serbia, 2024, pp. 209-212