Kauno technologijos universitetas

Informatikos fakultetas

Išmaniojo oro vėsintuvo-šildytuvo apsaugos nuo atvirkštinės inžinerijos analizė ir tyrimas

Baigiamasis magistro studijų projektas

Aivaras Chomskis
Projekto autorius

Doc. Šarūnas Packevičius
Projekto vadovas

Kaunas, 2024

Kauno technologijos universitetas

Informatikos fakultetas

Išmaniojo oro vėsintuvo-šildytuvo apsaugos nuo atvirkštinės inžinerijos analizė ir tyrimas

Baigiamasis magistro studijų projektas

IFM 2/2 gr. Aivaras Chomskis
Studentas

Doc. Šarūnas Packevičius
Projekto vadovas

Prof. Tomas Blažauskas
Recenzentas

Kaunas, 2024

Kauno technologijos universitetas

Informatikos fakultetas

Išmaniojo oro vėsintuvo-šildytuvo apsaugos nuo atvirkštinės inžinerijos analizė ir tyrimas

Akademinio sąžiningumo deklaracija

Patvirtinu, kad:

1. baigiamąjį projektą parengiau savarankiškai ir sąžiningai, nepažeisdama(s) kitų asmenų autoriausar kitų teisių, laikydamasi(s) Lietuvos Respublikos autorių teisių ir gretutinių teisių įstatymo nuostatų, Kauno technologijos universiteto (toliau – Universitetas) intelektinės nuosavybės valdymo ir perdavimo nuostatų bei Universiteto akademinės etikos kodekse nustatytų etikos reikalavimų;

2. baigiamajame projekte visi pateikti duomenys ir tyrimų rezultatai yra teisingi ir gauti teisėtai, neiviena šio projekto dalis nėra plagijuota nuo jokių spausdintinių ar elektroninių šaltinių, visos baigiamojo projekto tekste pateiktos citatos ir nuorodos yra nurodytos literatūros sąraše;

3. įstatymų nenumatytų piniginių sumų už baigiamąjį projektą ar jo dalis niekam nesu mokėjęs (- usi);

4. suprantu, kad išaiškėjus nesąžiningumo ar kitų asmenų teisių pažeidimo faktui, man bus taikomosakademinės nuobaudos pagal Universitete galiojančią tvarką ir būsiu pašalinta(s) iš Universiteto, o baigiamasis projektas gali būti pateiktas Akademinės etikos ir procedūrų kontrolieriaus tarnybai nagrinėjant galimą akademinės etikos pažeidimą.

Aivaras Chomskis

Patvirtinta elektroniniu būdu

Kauno technologijos universitetas

Informatikos fakultetas

Baigiamojo magistro projekto užduotis

Projekto tema	Išmaniojo oro vėsintuvo-šildytuvo apsaugos nuo atvirkštinės inžinerijos analizė ir tyrimas
Reikalavimai ir sąlygos (tikslinti pavadinimą pagal poreikį)	
Vadovas	Docentas Šarūnas Packevičius (vadovo pareigos, vardas, pavardė, parašas) 2024-05-14 (data)

Chomskis Aivaras. Išmaniojo oro vėsintuvo-šildytuvo apsaugos nuo atvirkštinės inžinerijos analizė ir tyrimas. Magistro baigiamasis projektas vadovas Doc. Šarūnas Packevičius; Kauno technologijos universitetas, Informatikos fakultetas.

Studijų kryptis ir sritis (studijų krypčių grupė): Programų sistemos

Reikšminiai žodžiai: apsauga, mikrovaldiklis, atvirkštinė

Kaunas, 2024. 65 p.

Santrauka

Šiame darbe aprašytas išmaniojo oro vėsintuvo-šildytuvo apsaugos nuo atvirkštinės inžinerijos analizė ir tyrimas. Programų sistemą sudaro išmanusis oro vėsintuvas-šildytuvas ir išorinio valdymo žiniatinklio programa. Sistemos paskirtis suteikti galimybę naudotojui valdyti oro vėsintuvą-šildytuvą išmaniaisiais būdais. Šie išmanieji būdai yra „Amazon Alexa“, „Google assistant“ ir žiniatinklio programą, kurioje naudotojas gali matyti duomenys ir valdyti patį įrenginį. Literatūros apžvalgoje analizuojama atvirkštinė inžinerija ir apsauga nuo atvirkštinės inžinerijos programinės įrangos inžinerijoje, elektronikos inžinerijoje, mechanikos inžinerijoje ir chemijos inžinerijoje. Tiriamojoje dalyje pateikiama programinio kodo statinė analizė ir galimi sistemos patobulinimai. Eksperimentinėje dalyje pateikiamos „ESP32“ mikrovaldiklio atminties pasisavinimo ir apsaugojimo procedūros ir palyginimas tarp apsaugoto ir neapsaugoto „ESP32“ mikrovaldiklio programos atminties turinio.

Chomskis Aivaras. Analysis and study of reverse engineering protection of smart air cooler – heater. Master‘s thesis project supervisor Doc. Šarūnas Packevičius; Kaunas University of Technology, Faculty of Informatics.

Field of study (group of fields of study): Software Engineering

Key words: protection, microcontroller, reverse

Kaunas, 2024. 65 p.

Summary

This paper describes the analysis and study of the reverse engineering protection of an intelligent air cooler-heater. The application system consists of an air smart cooler-heater and an external control web application. The purpose of the system is to enable the user to control the air cooler/heater in a smart way. These smart ways are Amazon Alexa, Google assistant and a web application where the user can see the data and control the unit itself. The literature review analyses reverse engineering and protection against reverse engineering in software engineering, electronic engineering, mechanical engineering and chemical engineering. The exploratory part presents a static analysis of the software code and possible improvements to the system. The experimental part presents the memory access and protection procedures for the ESP32 microcontroller and a comparison between the protected and unprotected contents of the ESP32 microcontroller program memory.

Turinys

Lentelių sąrašas

Paveikslų sąrašas

Santrumpų ir terminų sąrašas

PCB (angl. *Printed circuit board*) – spausdinta schema

IR – infraraudonieji spinduliai

ESP32 – mikrovaldiklis su įdiegtu „WiFi“ ir „Bluetooth“ ryšiu

DHT22 – temperatūros ir drėgmės jutiklis

API (angl. *application programming interface*) – aplikacijų programavimo sąsaja

IC (angl. *integrated circuit*) – integrinis grandynas

PA – panaudojjimo atvejis

FR – funkcinis reikalavimas

NFR – nefunkcinis reikalavimas

Įvadas

Išmanieji oro vėsintuvai ir šildytuvai tampa vis populiaresni, nes jie suteikia patogų ir efektyvų būdą reguliuoti temperatūrą namuose. Šie įrenginiai dažnai turi sudėtingą programinę ir aparatinę įrangą, kuri gali būti pažeidžiama atvirkštinei inžinerijai. Atvirkštinė inžinerija – tai procesas, kai iš sukurto produkto išgaunama informacija apie jo dizainą, veikimą ir gamybos procesą. Ši informacija gali būti naudojama įvairiems tikslams, pavyzdžiui, neteisėtai nukopijuoti produktui, rasti saugumo spragų.

Darbo aktualumas

Atvirkštinė inžinerija gali sukelti rimtų problemų vartotojams, nes padirbti išmanieji oro vėsinimo ir šildymo įrenginiai gali:

- būti nesaugūs ir kelti gaisro pavojų;
- neturėti visų originalių funkcijų arba veikti netinkamai;
- pagražinti energijos vartojimo efektyvumą, klaidinant vartotojus;
- pažeisti intelektinės nuosavybės teises.

Darbo tikslas

Šio tyrimo tikslas išanalizuoti ir įvertinti išmaniojo oro vėsintuvo-šildytuvo apsaugos nuo atvirkštinės inžinerijos mechanizmus ir patobulinti įrenginį apsaugant jį nuo atvirkštinės inžinerijos.

Uždaviniai

1. atlikti apsaugos nuo atvirkštinės inžinerijos įrankių analizę;
2. pasiūlyti patobulinimus ir apsaugos priemones išmaniajam oro vėsintuvui-šildytuvui siekiant, padidinti atsparumą atvirkštinei inžinerijai;
3. įdiegti pasiūlytus patobulinimus;
4. išbandyti ir ištirti pasiūlytus apsaugos mechanizmus.

Dokumento struktūra

Analitinėje dalyje pateikiama medžiaga apie atvirkštinę inžineriją ir apsaugą nuo jos. Projektinės dalies skyriuje pateikiami programų sistemos panaudojimo atvejai, reikalavimų specifikacija bei architektūros aprašas. Tiriamojoje dalyje pateikiama statinė sistemos kodo analizė. Eksperimentinėje dalyje pateikiamas tyrimas, kuriame aprašyta hipotezė, tyrimo eiga, rezultatai bei kita su tyrimu susijusi informacija.

1. Analitinė dalis

1.1. Atvirkštinė inžinerija

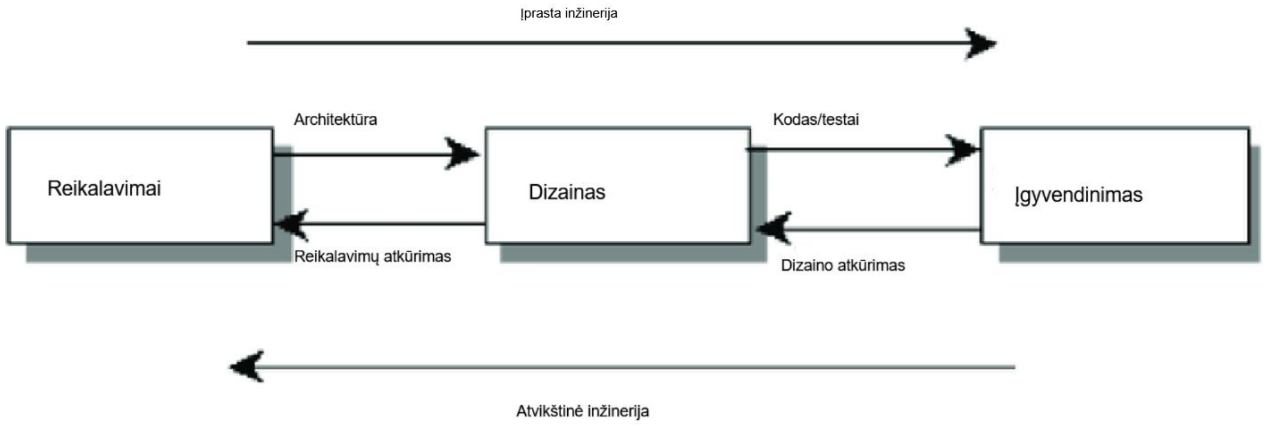
Atvirkštinė inžinerija yra procesas, kurio metu iš jau pagaminto objekto, įrenginio, programinės įrangos ar proceso bandoma suprasti jo veikimo principą, dizainą ir gamybos technologijas. Tai yra kaip dėlionės dėliojimas atvirkščiai: turint galutinį produktą, siekiama atskleisti jo vidinę struktūrą ir veikimo mechanizmus.

1.2. Atvirkštinės inžinerijos sritys

Atvirkštinė inžinerija gali būti taikoma įvairiose srityse. Atvirkštinė inžinerija yra taikoma programinės įrangos inžinerijoje, elektronikos inžinerijoje, mechanikos inžinerijoje, chemijos inžinerijoje.

1.2.1. Atvirkštinės inžinerija programinės įrangos inžinerijoje

Atvirkštinė inžinerija yra įprasta praktika kompiuterių aparatinės ir programinės įrangos srityje [1]. Naudojant atvirkštinę inžineriją siekiama patobulinti gaminio funkcijas arba pašalinti trūkumus. 1 paveiksle pavaizduota kuo skiriasi įprasta programinės įrangos inžinerija nuo atvirkštinės inžinerijos. Matome, kad įprastos inžinerijos atveju visi procesai eina iki galutinio įgyvendinimo etapo, o atvirkštinės inžinerijos atveju viskas atvirkščiai.



1 pav. Atvirkštinės ir normalios inžinerijos procesas [2]

1.2.2. Atvirkštinės inžinerijos programinės įrangos taikymas inžinerijoje

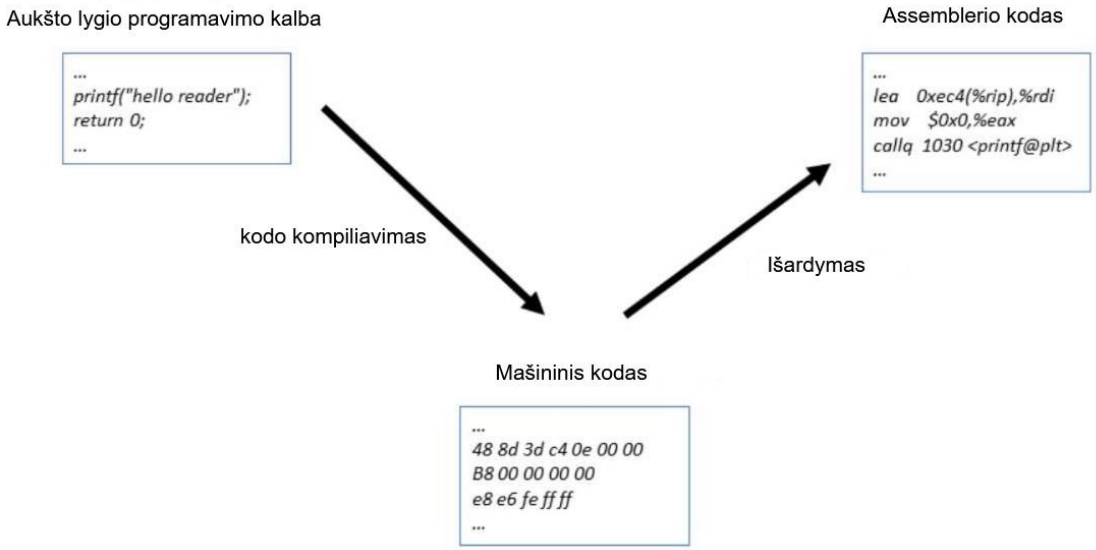
Atvirkštinė inžinerija taikoma, kai būtina suprasti ar modifikuoti esamą programinę įrangą. Štai keli jos taikymo atvejai:

- norint suprasti sistemos veikimą, kai trūksta senų sistemų arba trečiųjų šalių programų dokumentacijos;
- susiduriant su programinės įrangos klaidomis ar problemomis, neturint prieigos prie pirminio kodo;
- sprendžiant sąveikos problemas, kai integruojama su uždaromis ar patentuotomis sistemomis.
- programos kodo atkūrimas praradimo atveju;
- kenkėjiškų programų analizė kibernetiniam saugumui užtikrinti;
- produkto tobulinimas, analizuojant konkurentų produktą.

1.2.3. Atvirkštinės inžinerijos programinės įrangos inžinerijoje procesas

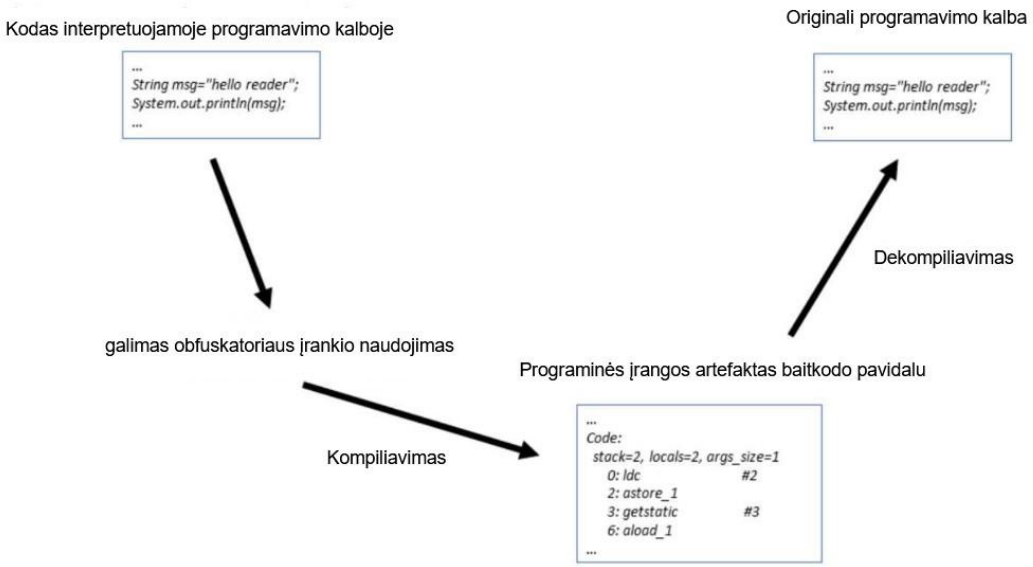
Atvirkštinės inžinerijos procesą galime suskirstyti į šešis žingsnius:

1. **Pasiruošimas** – pasirenkame tiriamąją programų sistemą. Nusprendžiame koks mūsų tyrimo tikslas, ar tai noras suprasti sistemą, ar tai sistemos klaidų taisymas. Įsitikiname ar teisiškai galime atlikti atvirkštinę inžineriją programinei įrangai. Surenkame informaciją apie tiriamąją programinę įrangą, pavyzdžiui, programinės įrangos dokumentacija, naudojimo vadovai. Išsirenkame įrankius su kuriais atliksime atvirkštinę inžineriją.
2. **Programinio kodo išgavimas ir sutvarkymas** – priklausomai nuo pačios programinės įrangos galime naudoti arba deasemblerį arba dekompiliatorių. Deasembleris paverčia mašininį kodą į žmogui suprantamą asemblerio kodą.



2 pav. Mašininio kodo išardymas [5]

Dekompiliatorius paverčia baitų kodą į aukšto lygio programavimo kalbą.



3 pav. Dekompiliavimo procedūra [5]

Taip yra palengvinamas darbas su sudėtingomis sistemomis ir taip išryškinama, kaip ir kodėl sistemos veikia [3].

3. **Statinė analizė** – programinės įrangos sistemos kodo tikrinimas jo nevykdant. Šis analizės būdas leidžia tyrėjams pamatyti, kaip kodas veikia, nustatyti pažeidžiamumus ir suprasti kodo struktūrą. Statinės analizės metu naudojamas programos kodas, kad būtų išgauta informacija apie kintamuosius, funkcijas, valdymo struktūras [4].
4. **Dinaminė analizė** – sistemos ar programinės įrangos elgsenos ir funkcionalumo nagrinėjimas jos veikimo metu. Dinaminė analizė nagrinėja, kaip sistema elgiasi realiuoju laiku. Šis etapas leidžia tyrėjams suprasti, kaip sąveikauja skirtingi komponentai, nustatyti pažeidžiamumus ar klaidas ir paslėptas funkcijas [4].
5. **Peržiūra** – jei atvirkštinės inžinerijos būdu sukurta programinė įranga neveikia taip, kaip norėtumėte, ji nebus naudinga. Atlikę programinės įrangos peržiūrą ir sutvarkę norimus dalykus būsite tikri, kad kodas veikia nuosekliai ir teisingai, be jokių klaidų ar trikdžių [3].
6. **Dokumentavimas** – galutinis žingsnis yra visų rezultatų, išvadų ir nustatytų ypatybių dokumentavimas. Tai svarbu ne tik kaip atsiskaitymo dalis, bet ir kaip žinių bazė ateities projektams ar panašiems tyrimams. Dokumentavimas turėtų apimti išsamius aprašymus, diagramas, lentelių pateikimus ir analizės metodų apibendrinimus.

1.2.4. Atvirkštinės inžinerijos įrankiai programinės įrangos inžinerijoje

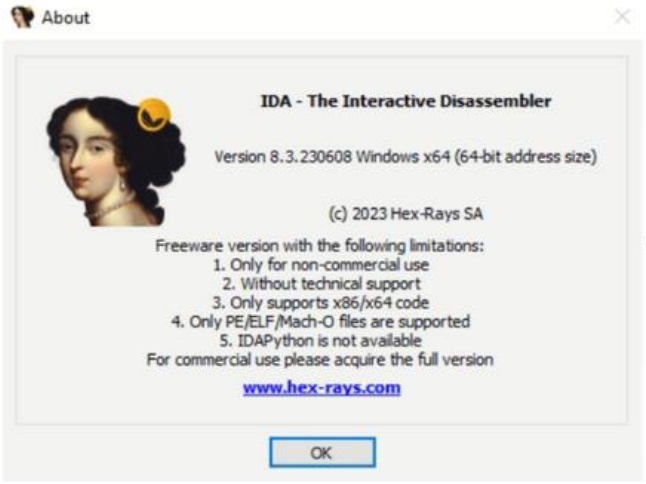
Norint atlikti atvirkštinę inžineriją programinės įrangos inžinerijos srityje reikia ir įrankių. Šiuos įrankius galima suskirstyti į keletą kategorijų: deasembleriai, dekompiliatoriai, derintuvės.

1.2.4.1. Deasembleriai (angl. *disassemblers*)

Deasembleriai yra labai svarbus atvirkštinės inžinerijos procese, nes jie leidžia analizuoti programinės įrangos veikimą gilesniu lygmeniu. Šie įrankiai skaito mašininį kodą, kurį kompiuterio procesorius tiesiogiai supranta ir vykdo, ir verčia jį į asemblerio kalbą, kuri yra žmogui suprantama teksto forma. Asemblerio kodas yra žemesnio lygio programavimo kalba ir artima mašininiam kodui, bet lengviau suprantama žmogui. Asemblerio kodas pateikia instrukcijas procesoriui žodžiais ir simboliais, kurie atitinka mašinines instrukcijas. Naudodami deasemblerius, inžinieriai gali pamatyti, kaip konkrečios programos arba jos dalys iš tikrųjų veikia procesoriaus lygmenyje.

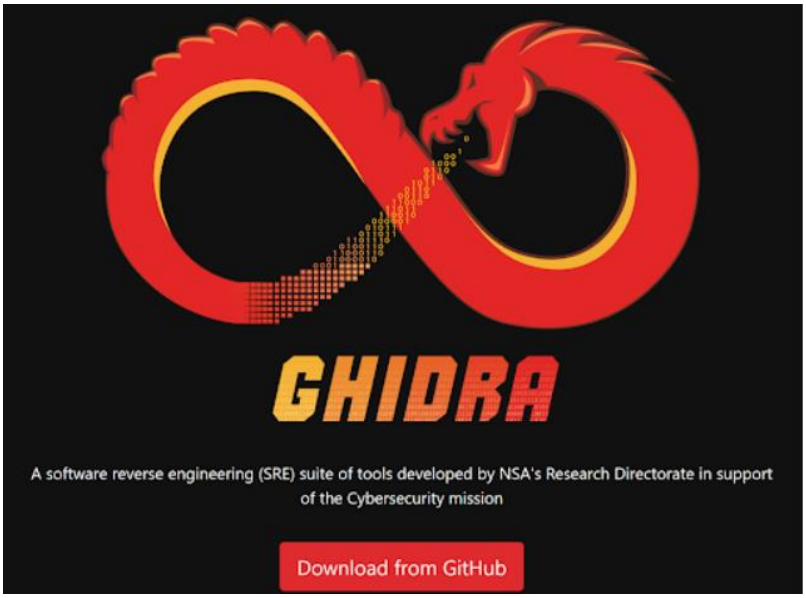
Keletas populiarių deasemblerių:

- **„IDA Pro“:** įmonės „Hex-Rays“ sukurta populiari programinė įranga, naudojama atvirkštinei dvejetainio kodo inžinerijai. Deasembleris ir kartu derinimo įrankis leidžia naudotojams gilintis į sudėtingas vykdomųjų failų detales, įskaitant vykdomąsias programas, bendrąsias bibliotekas, objektų failus ir programinę įrangą. Programa „IDA Pro“ palaiko įvairias procesorių architektūras ir failų formatus, todėl programa „IDA Pro“ yra universali įvairių platformų kodo analizės priemonė [6].



4 pav. „IDA Pro“ programa [6]

– **„Ghidra“:** atvirkštinės inžinerijos įrankis, kurį sukūrė „NSA“ ir kuris buvo išleistas 2019 metais. Šis įrankis tapo populiarus tarp kenkėjiškų programų analitikų, nes tai yra deasembleris. Tai leidžia kenkėjiškų programų analitikui patikrinti kenkėjiškos programos pavyzdžio funkcionalumą jo nepaleidžiant, tai labai naudinga, nes analitikas gali peržiūrėti kenkėjiškos programos kodą ir nustatyti, ką ji daro.



5 pav. „Ghidra“ programa [7]

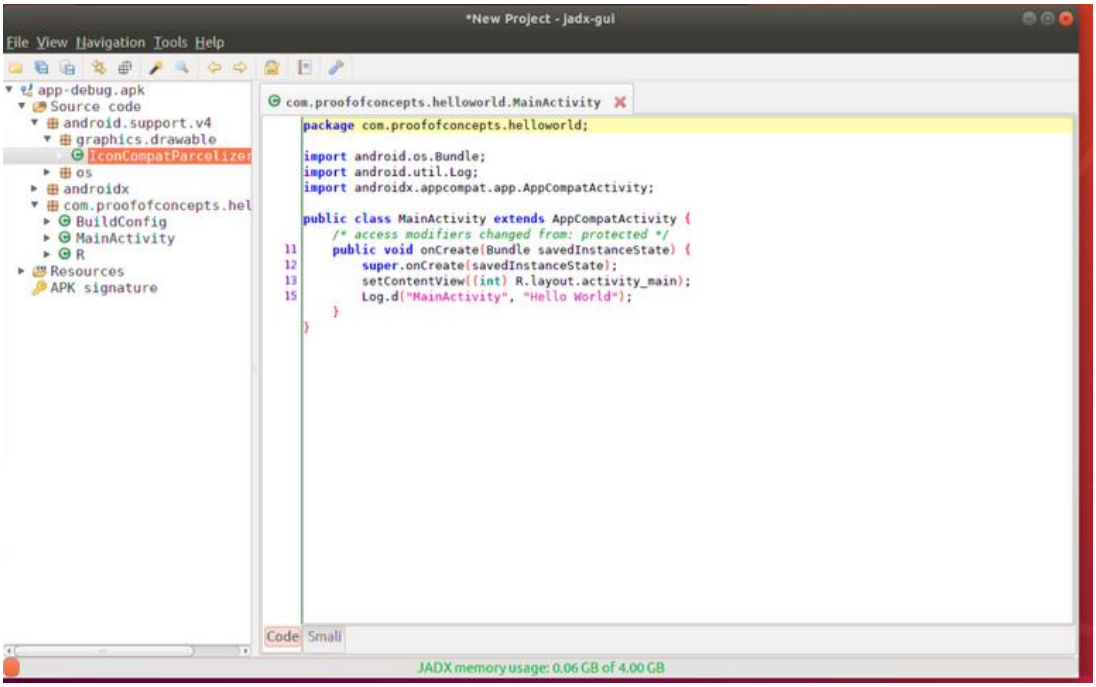
– **„Hopper Disassembler“:** „Hopper“ yra atvirkštinės inžinerijos įrankis, skirtas „macOS“ ir „Linux“, leidžiantis vartotojui išardyti ir dekompiliuoti 32 arba 64 bitų „Intel“ pagrindu veikiančias „Mac“, „Linux“, „Windows“ ir „iOS“ vykdomąsias programas [11].

1.2.4.2. Dekompiliatoriai (angl. *decompilers*)

Dekompiliatoriai yra dar viena svarbi atvirkštinės inžinerijos įrankių kategorija. Jos paverčia mašininį kodą ne tik į žemesnio lygio asemblerio kodą, bet ir bando atkurti aukštesnio lygio programavimo kalbą, pavyzdžiui, „C“, „C++“, „Java“ arba „Python“. Tai leidžia programuotojams lengviau suprasti ir analizuoti programinę įrangą, nes jie gali dirbti su aukštesnio lygio programavimo kalba, su kuria jie yra labiau įpratę dirbti.

Keletas populiarių dekompiliatorių:

– **„JADX“:** atvirojo kodo „Android“ programų dekompiliavimo įrankis. „JADX“ naudojamas „DEX/Smali“ kodui konvertuoti į „Java“ kodą [8].

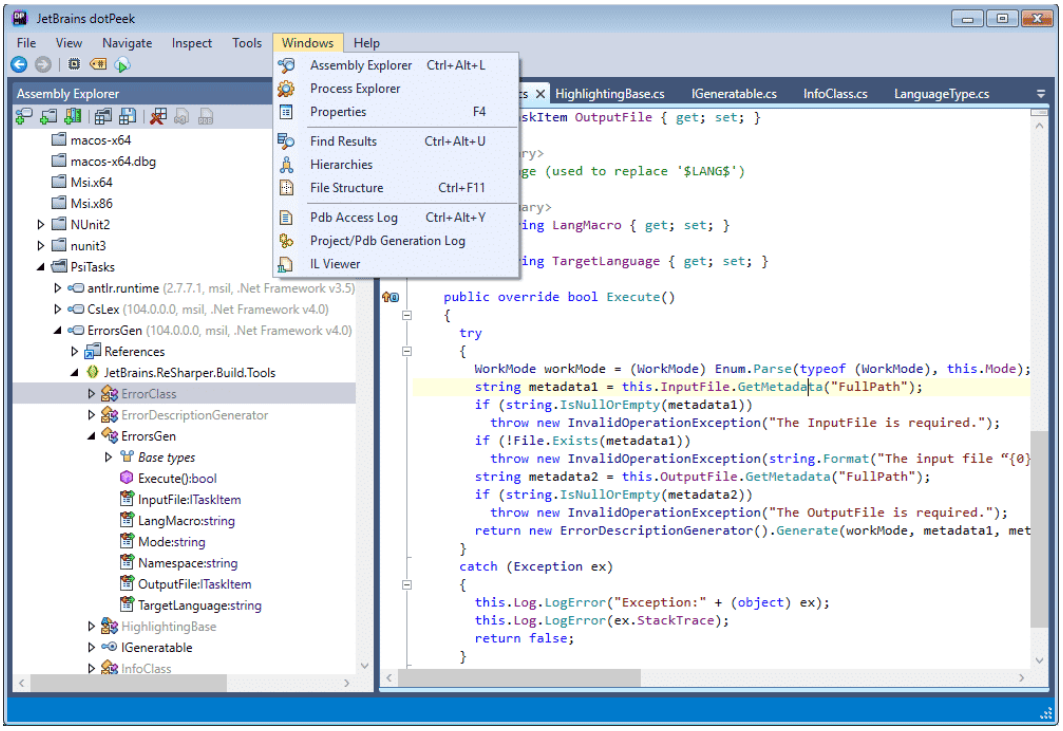


6 pav. „JADX“ grafinė sąsaja [8]

– **„.NET Reflector“:** tai yra dekompiliatorius ir klasių naršyklė, padedanti programų kūrėjams nagrinėti „.NET“ rinkinius ir metodus nuo pat „Microsoft .NET“ sistemos pradžios. Šis įrankis leidžia atidaryti „.DLL“ arba „.EXE“ failus ir atkurti jų „C#“ arba „Visual Basic“ kodą. Iš pradžių nemokamą įrankį sukūrė Lutzas Roederis 2008 metais. „.NET Reflector“ nusipirko „Red Gate Software Ltd.“. Dėl „Microsoft .NET“ pokyčių apimties ir spartos vienas programuotojas, dirbantis ne visą darbo dieną, greičiausiai negalėtų nuolat atnaujinti „.NET Reflector“, nekalbant jau apie tai, kad jį būtų galima išplėsti naujais būdais [9].

7 pav. „.NET Reflector“ logotipas [10]

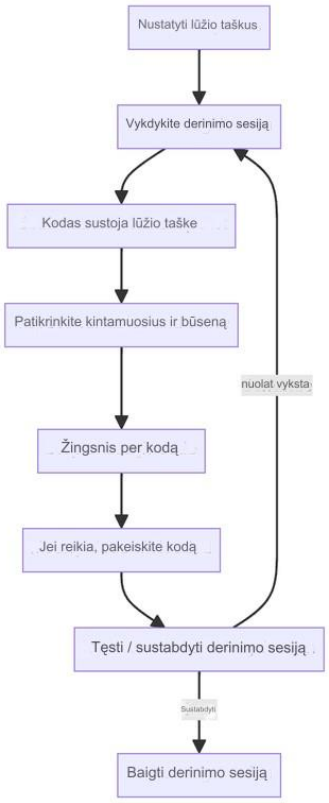
– **„DotPeek“:** tai nemokamas įrankis „.NET“ programų dekompiliavimui. Ši įrankis konvertuoja „.NET“ rinkinius į lygiavertį „C#“ kodą ir pasirinktinai rodo pagrindinį „IL“ kodą. „DotPeek“ įrankis gali dekompiliuoti įvairių rūšių asambliažų failus, pavyzdžiui, bibliotekas (.dll), vykdomuosius failus (.exe), „Windows 8“ metaduomenų failus (.winmd), archyvus (.zip), „NuGet“ paketus (.nupkg) ir „Microsoft Visual Studio Extensions“ paketus (.vsix) [12].



8 pav. „.DotPeek“ pagrindinis langas [12]

1.2.4.3. Derintuvės (angl. *debuggers*)

Derintuvės yra trečioji esminė atvirkštinės inžinerijos įrankių kategorija. Šios programos leidžia programuotojams vykdyti programas žingsnis po žingsnio, stabdyti programų veikimą tam tikrose vietose (vadinamose lūžio taškais), ir peržiūrėti bei keisti jų atmintyje esančias reikšmes. Tai suteikia giluminį supratimą apie tai, kaip programa veikia, leidžia nustatyti klaidas ir suprasti programinės įrangos veikimo principus. Derinimo programos yra ypač svarbios, kai reikia analizuoti programinės įrangos elgesį vykdymo metu, ypač jei šaltinio kodas nėra prieinamas arba yra sudėtingas. Jos padeda atlikti atvirkštinę inžineriją, identifikuojant programos veikimo logiką, klaidas ir galimas saugumo spragas.



9 pav. Programos derinimo sesija [15]

Populiarios derintuvės:

- **GDB (GNU Debugger**): Šis įrankis yra vienas populiariausių „UNIX“ ir „Linux“ šeimos sistemų derinimo įrankiu. Šis įrankis skirta programoms derinti ir tirti. Jis padeda suprasti kas vyksta programos viduje jos vykdymo metu. Jį galima įsigyti pagal „GNU“ projektą. „GDB“ naudojamas įvairiomis programavimo kalbomis parašytoms programoms, įskaitant „C“, „C++“, „D“, „Go“, „Java“, „Fortran“, „Pascal“ derinti [13].
- **PDB (Python Debugger)**: „Python“ derinimo įrankis, arba „PDB“ modulis, yra integruota „Python“ standartinės bibliotekos funkcija, kuri palengvina „Python“ programos derinimą. Jis nurodomas kaip klasė „Pdb“, kuri viduje naudoja „cmd“ (į eilutę orientuotų komandų interpretatorių palaikymas) ir „BDB“ (pagrindinės derinimo įrankio funkcijos) modulius. Pagrindinis „Pdb“ privalumas yra tas, kad ji veikia tik komandinėje eilutėje, todėl idealiai tinka kodo derinimui nutolusiuose serveriuose, kuriose nėra grafinės sąsajos. [14].

- **LLDB**: ši derintuvė yra numatytoji „Xcode“ naudojama derinimo programa, kuri yra „LLVM“ projekto dalis. „LLDB“ pakeitė „GDB XCode 5" ir turi keletą privalumų, pavyzdžiui, yra našesnė ir su ją galima kurti derinimo scenarijus [16].

1.2.5. Atvirkštinės inžinerija elektronikos inžinerijoje

Elektronikos atvirkštinė inžinerija – tai sistemingas elektroninio prietaiso ar sistemos dizaino, architektūros ir veikimo analizės supratimo procesas neturint prieigos prie originalių projektavimo specifikacijų, schemų ar pirminio kodo. Ji apima prietaiso išardymą, tikrinimą ir bandymą, siekiant surinkti informaciją apie jo komponentus, grandines ir programinę įrangą [17].

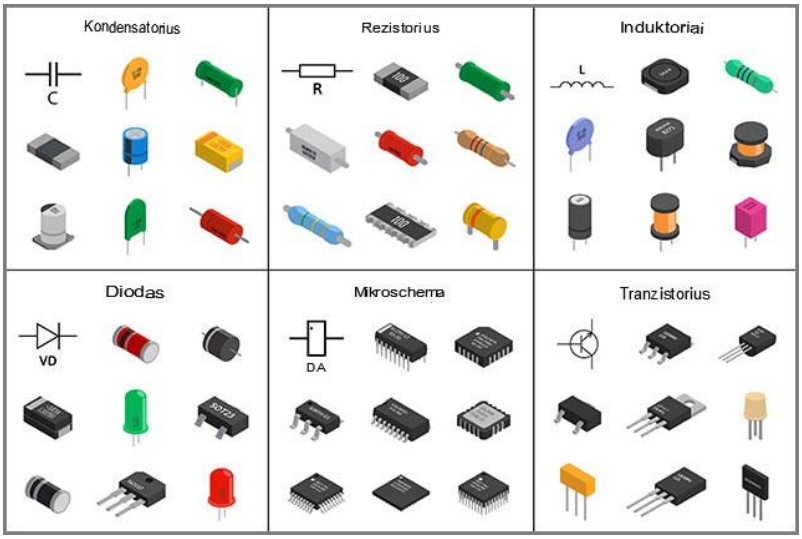
1.2.6. Atvirkštinės inžinerijos taikymas elektronikos inžinerijoje

- **Produktų tobulinimas:** gamintojai gali naudoti atvirkštinę inžineriją, kad išanalizuotų konkurentų elektronikos produktus ir rastų būdų, kaip pagerinti savo produktų dizainą, funkcionalumą ar našumą.
- **Gedimų šalinimas:** atvirkštinė inžinerija gali būti naudojama gedimų šalinimui sugedusiuose elektroniniuose prietaisuose, kai originalios schemos ar dokumentacijos nėra prieinamos.
- **Klastočių atpažinimas:** atvirkštinė inžinerija gali būti naudojama klastotų elektroninių prietaisų identifikavimui, palyginant juos su originaliais produktais.
- **Istorinis tyrimas:** atvirkštinė inžinerija gali būti naudojama senovinių elektroninių prietaisų veikimo principams suprasti.
- **Švietimas:** atvirkštinė inžinerija gali būti naudojama kaip edukacinis įrankis, siekiant studentams parodyti, kaip veikia elektroniniai prietaisai.

1.2.7. Atvirkštinės inžinerijos procesas elektronikos inžinerijoje

Visą atvirkštinės inžinerijos procesą galima suskirstyti į keletą etapų.

1. **Pasiruošimas**: pasirenkame prietaisą. Įsitikiname ar teisiškai galime atlikti atvirkštinę inžineriją pasirinktam prietaisui. Surenkame informaciją apie tiriamąjį prietaisą, pavyzdžiui, serviso dokumentaciją, naudojimo vadovus. Išsirenkame įrankius su kuriais atliksime atvirkštinę inžineriją.
2. **Išardymas:** elektroninis prietaisas yra išardomas, kad būtų galima apžiūrėti jo komponentus. Tai reiškia, kad reikia atsargiai išsukti varžtus, atpalaiduoti spaustukus ir kitas prietaisą laikančias tvirtinimo detales. Prietaisas gali būti itin sudėtingas, todėl svarbu dokumentuoti išardymo procesą, fotografuojant arba filmuojant kiekvieną žingsnį, kad būtų užtikrinta, jog prietaisą bus galima teisingai surinkti iš naujo [17].
3. **Komponentų identifikavimas:** šiame žingsnyje nustatome komponentus, kurie yra sumontuoti ant PCB plokštės. Šių plokščių gali būti viena arba daugiau. Identifikuojame atskiras mikroschemas, rezistorius, kondensatorius ir kitus įrenginį sudarančius komponentus.



10 pav. Elektronikos komponentai [19]

4. **Elektrinės grandinės analizė ir schemos atkūrimas:** grandinės schema yra atkuriama arba naudojant rastą dokumentaciją, arba rankinių būdu sekant ir dokumentuojant komponentus ir juos jungiančius takus. Šiuo proceso metu gali tekti ir nulituoti komponentus nuo plokštės.
5. **Programinės įrangos analizė:** programinė įranga, kurią naudoja prietaisas, yra analizuojama naudojant panašius ir tuos pačius įrankius, kuriuose naudojame ir programinės įrangos atvirkštinėje inžinerijoje. Tai tokie įrankiai kaip dekompiliatoriai, išardymo programos. Taip pat naudojami ir mikroschemų skaitytuvai ir integrinių grandynų programatoriai (IC) mikrovaldiklių programinei įrangai ar atminties turiniui išgauti ir analizuoti [18].
6. **Dokumentavimas:** surinkta tyrimo metu informacija yra dokumentuojama ataskaitų ir schemų pavidalu.

1.2.8. Atvirkštinės inžinerijos įrankiai elektronikos inžinerijoje

Atvirkštinės inžinerijos įrankius galime suskirstyti į keletą rūšių:

1. Mechaninio išardymo įrankiai

Pincetai, replės, atsuktuvai, lituokliai ir mikroskopai naudojami kruopščiai išardyti įrenginį ir išmontuoti ir išlituoti elektroninius komponentus iš spausdintinių plokščių.

2. Mikroschemų skaitytuvai ir programatoriai

Mikroschemų skaitytuvai ir programatoriai naudojami integrinių grandynų (IC) ir mikrovaldiklių programinei įrangai ar atminties turiniui išgauti ir analizuoti.

3. Matavimo įrankiai

Loginiai analizatoriai ir oscilografai yra būtini atitinkamai skaitmeniniams ir analoginiams signalams fiksuoti ir analizuoti, leidžiantys tirti grandinės elgseną ir ryšio protokolus.

4. Tinklo analizatoriai ir protokolų šnipinėjimo prietaisai

Tinklo analizatoriai ir protokolų analizatoriai naudojami tinklo srautui fiksuoti ir analizuoti, todėl galima suprasti ryšių protokolus ir duomenų formatus.

5. Dekompiliatoriai ir išardymo įrankiai

Išardymo įrankis paverčia mašininį kodą į žmogui suprantamą asemblerio kodą. Dekompiliavimo įrankis paverčia baitų kodą į aukšto lygio programavimo kalbą.

1.2.9. Atvirkštinė inžinerija mechanikos inžinerijoje

Atvirkštinė inžinerija mechanikos inžinerijoje yra procesas, kurio metu išanalizavus jau pagamintą gaminį, sukuriamas jo brėžinys ar modelis. Šis procesas leidžia suprasti gaminio veikimą, iš ko jis padarytas ir kaip jis pagamintas, o tai gali būti naudinga dalių keitimui, produktų kopijavimui, naujų gaminių kūrimui ar gamybos proceso tobulinimui. Atvirkštinė inžinerija gali būti įgyvendinama rankiniu būdu matuojant detales mikrometrais ir slankmačiais, tačiau dažniausiai pasitelkiamas „3D“ skenavimas ar kompiuterinė tomografija, o gauti duomenys apdorojami specialia programine įranga. Nors atvirkštinė inžinerija yra sudėtingas procesas, ji gali padėti sutaupyti laiko ir pinigų, pagerinti gaminių kokybę bei sukurti inovatyvius sprendimus [21].

11 pav. Atvirkštinės inžinerijos procesas mechanikos inžinerijoje [20]

1.2.10. Atvirkštinė inžinerija chemijos inžinerijoje

Atvirkštinė inžinerija chemijoje yra siekis atskleisti cheminės medžiagos ar produkto paslaptis. Šis procesas apima įvairių metodų, skirtų cheminės sudėties, struktūros ir savybių analizei, panaudojimą. Atvirkštinės inžinerijos specialistai naudoja spektroskopiją, chromatografiją ir mikroskopiją, kad išgautų kuo daugiau informacijos apie tiriamą objektą. Spektroskopija leidžia identifikuoti cheminius junginius, kurie sudaro medžiagą ir išsiaiškinti jų tarpusavio ryšius. Chromatografija padeda atskirti šiuos junginius ir išmatuoti junginių koncentracijas. Mikroskopija suteikia detalią vaizdinę

informaciją apie medžiagos struktūrą, atskleisdama jos vidinę sandarą ir mikroelementų išsidėstymą [22].

Atvirkštinė inžinerija chemijoje gali būti naudojama įvairiems tikslams, pavyzdžiui:

- **Naujų produktų kūrimas:** atvirkštinės inžinerijos dėka galima išanalizuoti jau sukurtus produktus ir sukurti panašius ar net geresnius variantus.
- **Medžiagų atpažinimas:** nežinomos kilmės medžiagos ar produkto sudėtį ir struktūrą galima nustatyti atliekant atvirkštinės inžinerijos tyrimą.
- **Teršalų tyrimas:** atvirkštinė inžinerija padeda identifikuoti ir analizuoti aplinkoje esančius cheminius teršalus.
- **Cheminių procesų supratimas:** išsamiai ištyrę cheminės medžiagos ar produkto savybes, galima geriau suprasti cheminius procesus, vykstančius jo viduje.

1.3. Apsauga nuo atvirkštinės inžinerijos skirtingose inžinerijos srityse

Inžinerijoje taikoma daug įvairių metodų norint apsisaugoti nuo atvirkštinės inžinerijos. Konkretūs naudojami metodai priklauso nuo inžinerijos srities ir apsaugoti norimos technologijos tipo. Apsauga nuo atvirkštinės inžinerijos gali būt taikoma programinės įrangos inžinerijoje, elektronikos inžinerijoje, mechanikos inžinerijoje, chemijos inžinerijoje.

1.3.1. Apsauga nuo atvirkštinės inžinerijos programinės įrangos inžinerijoje

Norint apsaugoti programinę įrangą nuo atvirkštinės inžinerijos yra keletas būdų kaip tai padaryti:

- **Kodo užmaskavimas:** šis metodas naudojamas siekiant programos kodą padaryti sunkiau suprantamą. Šis metodas apima kintamųjų ir funkcijų pervadinimą, siekiant paslėpti jų paskirtį, nereikalingų kodo šakų pridėjimą ir kodo blokų pertvarkymą. Nors kodo užmaskavimas nesustabdo pilnai atvirkštinės inžinerijos proceso, bet jis gali jį labai pristabdyti [23].
- **Kodo Šifravimas:** taikant šį metodą užšifruojamos jautrios kodo ar duomenų dalis, kad atvirkštinės inžinerijos specialistai negalėtų jų lengvai pasiekti. Dešifravimo raktai turėtų būti saugomi saugiai, o ne užkoduoti programinėje įrangoje**.** Tokiu būdu, net jei atvirkštinės inžinerijos specialistai gali išgauti užšifruotus duomenis, be iššifravimo rakto jų perskaityti neįmanoma [23].
- **Licencijų valdymas:** taikant šį metodą įdiegiamas patikimas licencijų valdymo ir programinės įrangos aktyvavimo sistemas. Taip užtikrinama, kad tik teisėti sistemos naudotojai galėtų naudotis visomis jūsų programinės įrangos funkcijomis. Licencijų tikrinimas ir aktyvavimas internetu gali apsunkinti inžinierių galimybes sukurti veikiančius nulaužimo ar raktų generatorius [23].
- **Apsauga nuo nenorimo sistemos derinimo:** Naudojant šį metodą siekiama įdiegti apsaugą nuo derintuvių, kurios gali labai aiškiai parodyti sistemos veikimą. Štai keletas taktikų:
    - **Funkcijų sąsaja**: „API“ sąsajos naudojimas yra labiausiai paplitęs ir tiesioginis statinis apsaugos nuo klaidų metodas. „Microsoft Windows“ operacinė sistema turi daug „API“ sąsajų, kurias galima naudoti siekiant nustatyti, ar yra naudojama derintuvė [24].

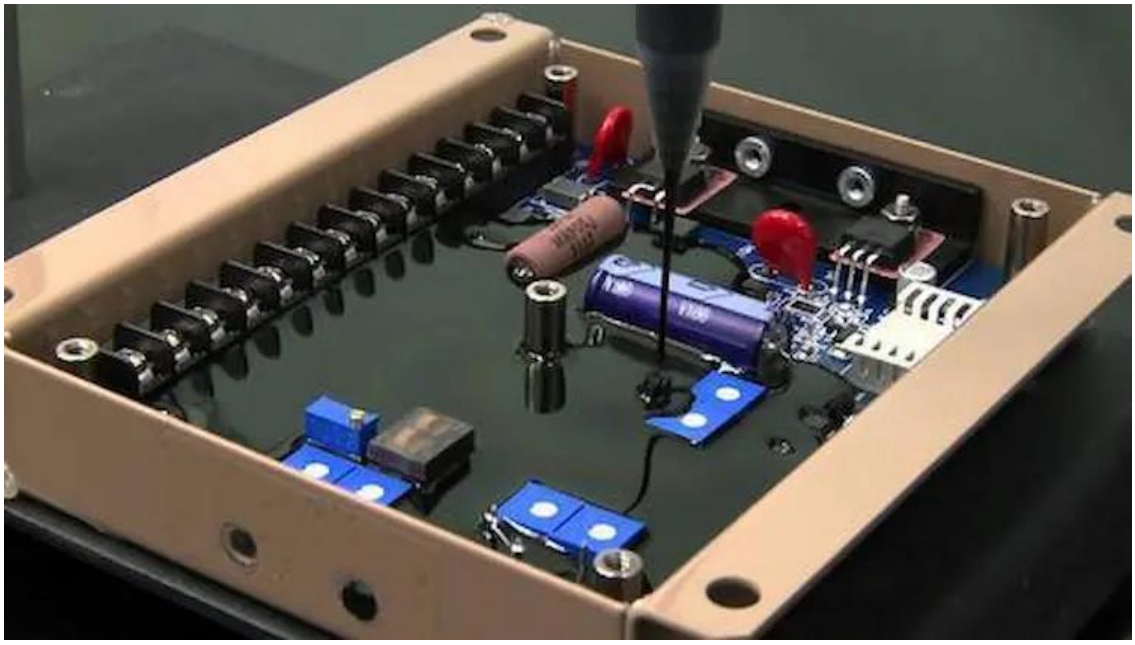
- **Lūžio taškų aptikimas**: lūžio taškų buvimas labai palengvina darbą derintuvei analizuojant programos vykdymo eigą, tačiau taip pat palieka svarbių įrodymų, patvirtinančių, kad egzistuoja derintuvė [24].
- **Valdymo srauto manipuliavimas** - tai tam tikrų priemonių naudojimas siekiant pakeisti arba paslėpti programos vykdymo giją, kad derintuvei būtų sunku ją surasti arba prie jos prisijungti [24].

– **Klastojimo aptikimas:** taikant šį metodą įtraukiami klastojimo aptikimo mechanizmai. Šie mechanizmai gali patikrinti vykdomojo failo vientisumą ir pažymėti bet kokius pakeitimus ar neleistinus pakeitimus. Aptikus klastojimą, programinė įranga gali atsisakyti paleisti arba imtis atitinkamų veiksmų [23].
– **Dinaminis kodo vykdymas:** taikant šį metodą suskirstomas programos kodas į kelias dalis ir dinamiškai įkeliamas paleidimo metu. Šis metodas pasunkina viso kodo bazės tyrinėjimą [23].

1.3.2. Apsauga nuo atvirkštinės inžinerijos elektronikos inžinerijoje

Norint apsaugoti elektronikos įrangą nuo atvirkštinės inžinerijos yra keletas būdų kaip tai padaryti:

– **Elektronikos dalių sandarinimas epoksidine plėvele:** taikant šį metodą visa PCB arba tam tikri svarbūs elektronikos komponentai padengiami epoksidinės dervos sluoksniu. Epoksidinis sandarinimas sukuria fizinį barjerą, kuris apsunkina priėjimą prie PCB, todėl apsunkinami bandymai atlikti atvirkštinę inžineriją [25].



12 pav. Elektronikos dalių sandarinimas epoksidine plėvele [26]

– **Integrinių grandynų ženklinimų pašalinimas:** taikant šį metodą nuo integruotųjų grandynų (IC), esančių ant PCB, pašalinami identifikavimo ženklai [25].
– **Aklųjų sujungimų naudojimas:** naudodami akluosius sujungimus PCB projekte, apsunkinamas grandinės išdėstymas, nes sujungimai tarp sluoksnių tampa ne tokie akivaizdūs [25].
– **Individualizuoto integrinio grandyno naudojimas:** pagal užsakymą sukurta integrinė schema, sukurta specialiai vienam produktui ir pritaikyta atlikti unikalią funkciją. Dėl tokio

sprendimo konkurentams sunkiau atkurti dizainą, nes jiems reikėtų kurti savo integrinį grandyną arba rasti nestandartinę alternatyvą, kuri gali būti ne tokio pat našumo ar funkcionalumo [25].

1.3.3. Apsauga nuo atvirkštinės inžinerijos mechanikos inžinerijoje

Norint apsaugoti mechaninę įrangą nuo atvirkštinės inžinerijos yra sukurta keletas būdų kaip tai padaryti:

- **Dizaino apsauga**: taikant šį metodą naudojami sudėtingi dizainai, kuriuos sunku nukopijuoti ar atkurti. Naudojamos patentuojamos technologijos ir komponentai. Slepiama kritiška informacija, pavyzdžiui, matmenis ir tolerancijos, kurie turėtų būti brėžiniuose ir specifikacijose.
- **Gamybos apsauga:** taikant šį metodą naudojami specialius gamybos procesai, kurie sunkiai atkartojami. Taip pat galime taikyti apsaugines dangas ar apdailas, apsunkinančias išardymą ir analizę.

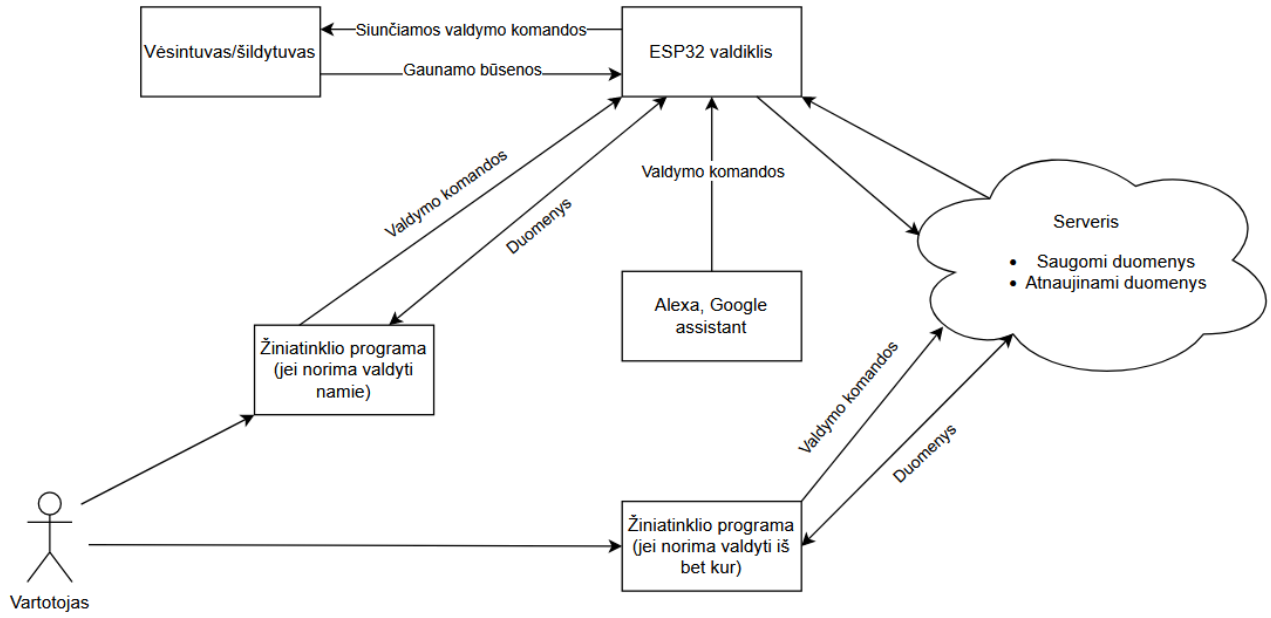
1.3.4. Apsauga nuo atvirkštinės inžinerijos chemijos inžinerijoje:

Norint apsaugoti chemijos produktą nuo atvirkštinės inžinerijos yra keletas būdų kaip tai padaryti:

- **Patentų registravimas:** patentuojant naujas chemines medžiagas, formules ar gamybos procesus, įmonės gali apsisaugoti nuo to, kad jų išradimai būtų legaliai nukopijuoti ar panaudoti konkurentų. Patentai suteikia teisę reikalauti teisinės apsaugos nuo pažeidėjų.
- **Slaptos informacijos apsauga:** naudodami konfidencialumo susitarimus su darbuotojais, tiekėjais ir partneriais, įmonės gali apsaugoti svarbią neviešą informaciją. Tai reiškia, kad informacija apie gamybos procesus, technologines paslaptis ar naujoves yra laikoma konfidencialia.

2. Projektinė dalis

2.1. Sistemos kontekstas



13 pav. Detali sistemos vizija

Produkto pagrindinės dalys yra vėsintuvas-šildytuvas, „ESP32“ mikrovaldiklis ir serveris.

Vartotojas gali atlikti valdymo pakeitimus panaudojant trys valdymo būdus: žiniatinklio programa naudojant namuose, žiniatinklio programa naudojant išvykus, „Amazon Alexa“ arba „Google assistant“. Taip pat galima naudoti ir gamykliškai įdiegtus valdymo būdus: Spaudant ant valdymo įrenginio esančius mygtukus arba naudojant valdymo pultelį.

Naudojant žiniatinklio programą iš namų nereikalingas interneto ryšys, nes prie įrenginio prisijungiama panaudojant „ESP32“ mikrovaldiklio skleidžiama prieigos tašką.

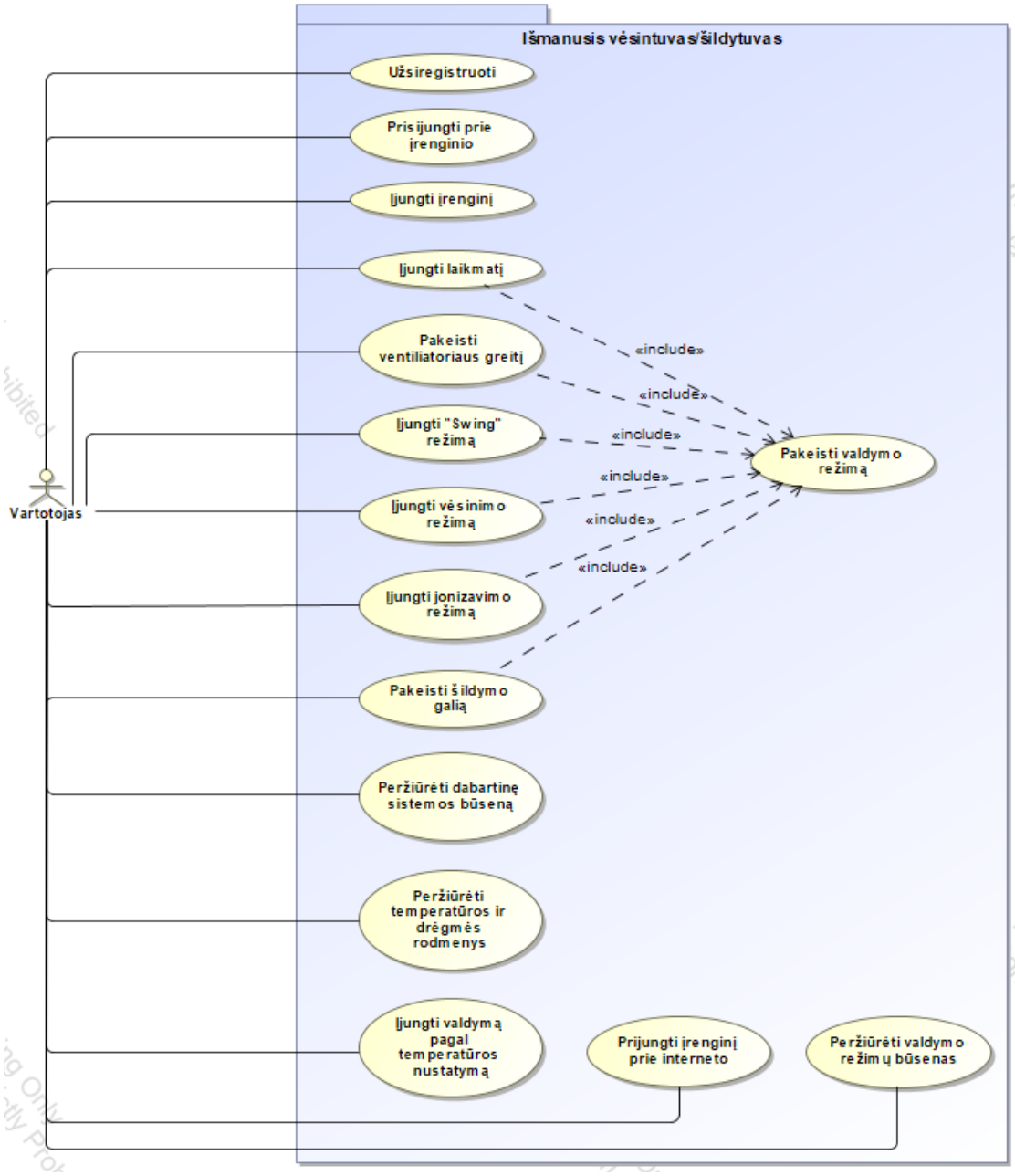
Serveryje bus kaupiama valdymo būsenos, temperatūros ir drėgmės rodmenys, prisijungimo duomenys.

Įrenginyje yra IR komunikaciją [27]. Šis komunkacijos būdas leis valdyti įrenginį naudojant IR siųstuvą. Dažniausiai sutinkamas IR protokolas yra „NEC“ [28].

2.2. Sistemos sudėtis (panaudojimo atvejų modelis)

2.2.1. Panaudojimo atvejų diagrama

Šioje sistemoje yra įgyvendinta 15 panaudojimo atvejų.



14 pav. Panaudojimo atvejų diagrama

2.2.2. Panaudojimo atvejai

1 lentelė. PA1 specifikacija

Panaudojimo atvejis	**Nr. 1**	**Užsiregistruoti**
Tikslas	Vartotojas turi užsiregistruoti sistemoje, kad galėtų valdyti įrenginį nebūnant namie.	
Dalyviai	Vartotojas	
Ryšiai su kitais PA	-	
Nefunkciniai reikalavimai	-	
Prieš-sąlygos	Atsidaryti registracijos langą	
Sužadinimo sąlyga	Suvedami duomenys ir paspaudžiamas registracijos mygtukas	
Po-sąlyga	Vartotojas pamato pavykusios registracijos pranešimą	
Pagrindinis scenarijus	Vartotojui suvedus duomenis ir paspaudus registracijos mygtuką, jo duomenys išsaugomi duomenų bazėje bei parodomas pavykusios registracijos pranešimas	
Alternatyvūs scenarijai	Jei blogi duomenys ar toks vartotojas jau užsiregistravęs, parodomas klaidos pranešimas	

2 lentelė. PA2 specifikacija

Panaudojimo atvejis	**Nr. 2**	**Prisijungti prie įrenginio**
Tikslas	Vartotojas turi prisijungti prie įrenginio, kad galėtų valdyti įrenginį	
Dalyviai	Vartotojas	
Ryšiai su kitais PA	-	
Nefunkciniai reikalavimai	-	
Prieš-sąlygos	Atsidaryti prisijungimo langą	
Sužadinimo sąlyga	Suvedami duomenys ir paspaudžiamas prisijungimo mygtukas	
Po-sąlyga	Vartotojas gali valdyti įrenginį	
Pagrindinis scenarijus	Vartotojui suvedus duomenis ir paspaudus prisijungimo mygtuką, vartotojui parodomas pagrindinis langas	
Alternatyvūs scenarijai	Jei įvesti blogi duomenys, vartotojui parodomas klaidos pranešimas	

3 lentelė. PA3 specifikacija

Panaudojimo atvejis	**Nr. 3**	**Įjungti įrenginį**
Tikslas	Vartotojas gali įjungti įrenginį	
Dalyviai	Vartotojas	
Ryšiai su kitais PA	-	
Nefunkciniai reikalavimai	NF7 Sistema turi atlikti būsenos pakeitimą neviršijant 300 sekundžių	
Prieš-sąlygos	Atsidaryti pagrindinį langą	
Sužadinimo sąlyga	Pagrindiniame lange paspausti įjungimo mygtuką arba pasakyti komandą „Amazon Alexa“ įrenginiui arba „Google Assistant“ virtualiam asistentui	
Po-sąlyga	Įrenginys veikia	
Pagrindinis scenarijus	Vartotojas atsidaro pagrindinį langą. Pagrindiniame lange paspaudžiamas įjungimo mygtukas arba pasakoma komanda „Amazon Alexa“ įrenginiui arba „Google Assistant“ virtualiam asistentui, įrenginys įjungiamas	
Alternatyvūs scenarijai	Jei nesuveikė norima komanda, rodoma sistemos klaida klaidų laukelyje	

4 lentelė. PA4 specifikacija

Panaudojimo atvejis	**Nr. 4**	**Pakeisti ventiliatoriaus greitį**
Tikslas	Vartotojas gali pakeisti ventiliatoriaus greitį	
Dalyviai	Vartotojas	
Ryšiai su kitais PA	Pakeisti valdymo režimą	
Nefunkciniai reikalavimai	NF7 Sistema turi atlikti būsenos pakeitimą neviršijant 300 sekundžių	
Prieš-sąlygos	Atsidaryti pagrindinį langą	
Sužadinimo sąlyga	Pagrindiniame lange paspausti ventiliatoriaus greičio keitimo	
Po-sąlyga	Pasikeičia ventiliatoriaus greitis	
Pagrindinis scenarijus	Vartotojas atsidaro pagrindinį langą. Pagrindiniame lange paspaudžia ventiliatoriaus greičio keitimo mygtuką. Pakeičiamas ventiliatoriaus greitis.	
Alternatyvūs scenarijai	Jei nesuveikė norima komanda, rodoma sistemos klaida klaidų laukelyje	

5 lentelė. PA5 specifikacija

Panaudojimo atvejis	**Nr. 5**	**Įjungti laikmatį**
Tikslas	Vartotojas gali įjungti laikmatį	
Dalyviai	Vartotojas	
Ryšiai su kitais PA	Pakeisti valdymo režimą	
Nefunkciniai reikalavimai	NF7 Sistema turi atlikti būsenos pakeitimą neviršijant 300 sekundžių	
Prieš-sąlygos	Atsidaryti pagrindinį langą	
Sužadinimo sąlyga	Pagrindiniame lange paspausti laikmačio įjungimo mygtuką	
Po-sąlyga	Įjungiamas laikmatis	
Pagrindinis scenarijus	Vartotojas atsidaro pagrindinį langą. Pagrindiniame lange paspaudžia laikmačio įjungimo mygtuką. Pakeičiama laikmačio būsena.	
Alternatyvūs scenarijai	Jei nesuveikė norima komanda, rodoma sistemos klaida klaidų laukelyje	

6 lentelė. PA6 specifikacija

Panaudojimo atvejis	**Nr. 6**	**Įjungti „Swing“ režimą**
Tikslas	Vartotojas gali įjungti „Swing“ režimą	
Dalyviai	Vartotojas	
Ryšiai su kitais PA	Pakeisti valdymo režimą	
Nefunkciniai reikalavimai	NF7 Sistema turi atlikti būsenos pakeitimą neviršijant 300 sekundžių	
Prieš-sąlygos	Atsidaryti pagrindinį langą	
Sužadinimo sąlyga	Pagrindiniame lange paspausti „Swing“ režimo	
Po-sąlyga	Įjungiamas „Swing“ režimas	
Pagrindinis scenarijus	Vartotojas atsidaro pagrindinį langą. Pagrindiniame lange paspaudžia „Swing“ režimo mygtuką. Pakeičiama „Swing“ režimo būsena.	
Alternatyvūs scenarijai	Jei nesuveikė norima komanda, rodoma sistemos klaida klaidų laukelyje	

7 lentelė. PA7 specifikacija

Panaudojimo atvejis	**Nr. 7**	**Įjungti vėsinimo režimą**
Tikslas	Vartotojas gali įjungti vėsinimo režimą	
Dalyviai	Vartotojas	
Ryšiai su kitais PA	Pakeisti valdymo režimą	
Nefunkciniai reikalavimai	NF7 Sistema turi atlikti būsenos pakeitimą neviršijant 300 sekundžių	
Prieš-sąlygos	Atsidaryti pagrindinį langą	
Sužadinimo sąlyga	Pagrindiniame lange paspausti vėsinimo režimo	
Po-sąlyga	Įjungiamas vėsinimo režimas	
Pagrindinis scenarijus	Vartotojas atsidaro pagrindinį langą. Pagrindiniame lange paspaudžia vėsinimo režimo mygtuką. Pakeičiama vėsinimo režimo būsena.	
Alternatyvūs scenarijai	Jei nesuveikė norima komanda, rodoma sistemos klaida klaidų laukelyje	

8 lentelė. PA8 specifikacija

Panaudojimo atvejis	**Nr. 8**	**Įjungti jonizavimo režimą**
Tikslas	Vartotojas gali įjungti jonizavimo režimą	
Dalyviai	Vartotojas	
Ryšiai su kitais PA	Pakeisti valdymo režimą	
Nefunkciniai reikalavimai	NF7 Sistema turi atlikti būsenos pakeitimą neviršijant 300 sekundžių	
Prieš-sąlygos	Atsidaryti pagrindinį langą	
Sužadinimo sąlyga	Pagrindiniame lange paspausti jonizavimo režimo mygtuką	
Po-sąlyga	Įjungiamas jonizavimo režimas	
Pagrindinis scenarijus	Vartotojas atsidaro pagrindinį langą. Pagrindiniame lange paspaudžia jonizavimo režimo mygtuką. Pakeičiama jonizavimo režimo būsena.	
Alternatyvūs scenarijai	Jei nesuveikė norima komanda, rodoma sistemos klaida klaidų laukelyje	

9 lentelė.PA9 specifikacija

Panaudojimo atvejis	**Nr. 9**	**Pakeisti šildymo galią**
Tikslas	Vartotojas gali pakeisti šildymo galią	
Dalyviai	Vartotojas	
Ryšiai su kitais PA	Pakeisti valdymo režimą	
Nefunkciniai reikalavimai	NF7 Sistema turi atlikti būsenos pakeitimą neviršijant 300 sekundžių	
Prieš-sąlygos	Atsidaryti pagrindinį langą	
Sužadinimo sąlyga	Pagrindiniame lange paspausti šildymo galio keitimo mygtuką	
Po-sąlyga	Pakeičiamas šildymo galios nustatymas	
Pagrindinis scenarijus	Vartotojas atsidaro pagrindinį langą. Pagrindiniame lange paspaudžia galio keitimo mygtuką. Pakeičiama šildymo galia.	
Alternatyvūs scenarijai	Jei nesuveikė norima komanda, rodoma sistemos klaida klaidų laukelyje	

10 lentelė. PA10 specifikacija

Panaudojimo atvejis	**Nr. 10**	**Peržiūrėti dabartinę sistemos būseną**
Tikslas	Vartotojas gali peržiūrėti dabartinę sistemos būseną	
Dalyviai	Vartotojas	
Ryšiai su kitais PA	-	
Nefunkciniai reikalavimai	-	
Prieš-sąlygos	Įsijungti žiniatinklio programa	
Sužadinimo sąlyga	Atsidaryti pagrindinį langą	
Po-sąlyga	Vartotojas gali matyti dabartinę sistemos būseną	
Pagrindinis scenarijus	Vartotojas atsidaro pagrindinį langą. Vartotojas atsidaro pagrindinį langą.Vartotojas mato dabartinę sistemos būseną.	
Alternatyvūs scenarijai	Vartotojas pakeitė režimą, bet būsena nepakito – rodoma klaida klaidų laukelyje	

11 lentelė. PA11 specifikacija

Panaudojimo atvejis	**Nr. 11**	**Peržiūrėti temperatūros ir drėgmės rodmenys**
Tikslas	Vartotojas peržiūrėti temperatūros ir drėgmės rodmenys	
Dalyviai	Vartotojas	
Ryšiai su kitais PA	-	
Nefunkciniai reikalavimai	-	
Prieš-sąlygos	Įsijungti žiniatinklio programą	
Sužadinimo sąlyga	Atsidaryti pagrindinį langą	
Po-sąlyga	Matomi temperatūros ir drėgmės rodmenys	
Pagrindinis scenarijus	Vartotojas įsijungia žiniatinklio programą. Vartotojas atsidaro pagrindinį langą. Vartotojas mato temperatūros ir drėgmės rodmenys	
Alternatyvūs scenarijai	Klaidų laukelyje matome temperatūros jutiklio gedimą	

12 lentelė. PA12 specifikacija

Panaudojimo atvejis	**Nr. 12**	**Įjungti valdymą pagal temperatūros nustatymą**
Tikslas	Vartotojas gali įjungti valdymą pagal temperatūros nustatymą	
Dalyviai	Vartotojas	
Ryšiai su kitais PA	-	
Nefunkciniai reikalavimai	-	
Prieš-sąlygos	Atsidaryti pagrindinį langą	
Sužadinimo sąlyga	Pagrindiniame lange paspaudžiamas valdymo pagal temperatūros nustatymą režimo mygtukas	
Po-sąlyga	Įjungiamas valdymas pagal temperatūros nustatymas	
Pagrindinis scenarijus	Vartotojas įsijungia pagrindinį langą. Pagrindiniame lange paspaudžiamas valdymo temperatūros nustatymą mygtuką	
Alternatyvūs scenarijai	Jei nesuveikia valdymo komanda – nepasikeičia temperatūros nustatymo būsena	

13 lentelė. PA13 specifikacija

Panaudojimo atvejis	**Nr. 13**	**Pakeisti valdymo režimą**
Tikslas	Vartotojas gali pakeisti valdymo režimą	
Dalyviai	Vartotojas	
Ryšiai su kitais PA	-	
Nefunkciniai reikalavimai	NF7 Sistema turi atlikti būsenos pakeitimą neviršijant 300 sekundžių	
Prieš-sąlygos	Atsidaryti pagrindinį langą	
Sužadinimo sąlyga	Pagrindiniame lange paspausti viena iš režimų mygtukų	
Po-sąlyga	Įjungiamas pasirinktas režimas	
Pagrindinis scenarijus	Vartotojas atsidaro pagrindinį langą. Pagrindiniame lange paspaudžia pasirinkto režimo mygtuką. Pakeičiama būsena.	
Alternatyvūs scenarijai	Jei nesuveikė norima komanda, rodoma sistemos klaida klaidų laukelyje	

14 lentelė. PA14 specifikacija

Panaudojimo atvejis	**Nr. 14**	**Prijungti įrenginį prie interneto**
Tikslas	Vartotojas gali prijungti įrenginį prie interneto	
Dalyviai	Vartotojas	
Ryšiai su kitais PA	-	
Nefunkciniai reikalavimai	NF4 Įrenginys turi būti lengvai prijungiamas prie interneto	
Prieš-sąlygos	Atsidaryti nustatymų langą	
Sužadinimo sąlyga	Randamas norimas prisijungimo taškas, suvedamas slaptažodis ir paspaudžiamas interneto prijungimo mygtukas	
Po-sąlyga	Įrenginys prijungiamas prie interneto	
Pagrindinis scenarijus	Vartotojas įsijungia nustatymų langą. Randamas norimas prisijungimo taškas, suvedamas slaptažodis ir paspaudžiamas interneto prijungimo mygtukas, vartotojui parodomas sėkmingo prisijungimo pranešimas	
Alternatyvūs scenarijai	Jei suvedami neteisingi duomenys, įrenginys neprisijungia prie interneto – parodomas klaidos pranešimas	

15 lentelė. PA15 specifikacija

Panaudojimo atvejis	**Nr. 15**	**Peržiūrėti valdymo režimų būsenas**
Tikslas	Vartotojas gali peržiūrėti valdymo režimų būsenas	
Dalyviai	Vartotojas	
Ryšiai su kitais PA	-	
Nefunkciniai reikalavimai	-	
Prieš-sąlygos	Įsijungti žiniatinklio programa	
Sužadinimo sąlyga	Atsidaryti pagrindinį langą	
Po-sąlyga	Vartotojas gali matyti valdymo režimų būsenas	
Pagrindinis scenarijus	Vartotojas įsijungia žiniatinklio aplikaciją. Vartotojas atsidaro pagrindinį langą.Vartotojas mato valdymo režimų būsenas	
Alternatyvūs scenarijai	Jei nesuveikė norima komanda, rodoma sistemos klaida klaidų laukelyje	

2.3. Funkciniai reikalavimai

Programų sistemoje yra du funkciniai reikalavimai.

16 lentelė. FR1 specifikacija

Reikalavimas #:	**FR1** **Reikalavimo tipas: V9**	**Įvykis/panaudojimo atvejis: 3-15**
Aprašymas:	Sistema turi informuoti apie įrangos gedimą	
Pagrindimas:	Vartotojas sužinos apie klaidą ir galės kreiptis į įrangos tiekėją.	
Šaltinis:	Sistemos užsakovas	
Tinkamumo kriterijus:	Sistema informuoja apie įrangos gedimą	
Užsakovo tenkinimas:	5	Užsakovo netenkinimas: 5
Priklausomybės:	Nėra	Konfliktai: Nėra
Papildoma medžiaga:		
Istorija:	Sukurtas: 2023-02-25	

17 lentelė. FR2 specifikacija

Reikalavimas #:	**FR2** **Reikalavimo tipas: V9**	**Įvykis/panaudojimo atvejis: 7**
Aprašymas:	Sistema turi informuoti apie žema vandens lygį bake	
Pagrindimas:	Vartotojas sužinos apie žemą vandens lygi bake ir galės ji papildyti	
Šaltinis:	Sistemos užsakovas	
Tinkamumo kriterijus:	Sistema informuoja apie žemą vandens lygi bake	
Užsakovo tenkinimas:	3	Užsakovo netenkinimas: 2
Priklausomybės:	Nėra	Konfliktai: Nėra
Papildoma medžiaga:		
Istorija:	Sukurtas: 2023-02-25	

2.4. Nefunkciniai reikalavimai

Programų sistemoje yra 16 nefunkcinių reikalavimų.

2.4.1. Atsparumas trukdžiams, klaidoms

18 lentelė. NF9 specifikacija

Reikalavimas #:	**NF9 Reikalavimo tipas: V12.5 Įvykis/panaudojimo atvejis: 14**		
Aprašymas:	Sistema turi pati automatiškai prisijungti prie interneto po įtampos dingimo ir atsiradimo		
Pagrindimas:	Sistema privalo automatiškai atsistatyti, kad nesukeltu vartotojui nepatogumų		
Šaltinis:	Sistemos užsakovas		
Tinkamumo kriterijus:	Sistema automatiškai prisijungia prie interneto po įtampos dingimo ir atsiradimo		
Užsakovo tenkinimas:	3	Užsakovo netenkinimas:	4
Priklausomybės:	NF4	Konfliktai:	Nėra
Papildoma medžiaga:			
Istorija:	Sukurtas: 2023-05-25		

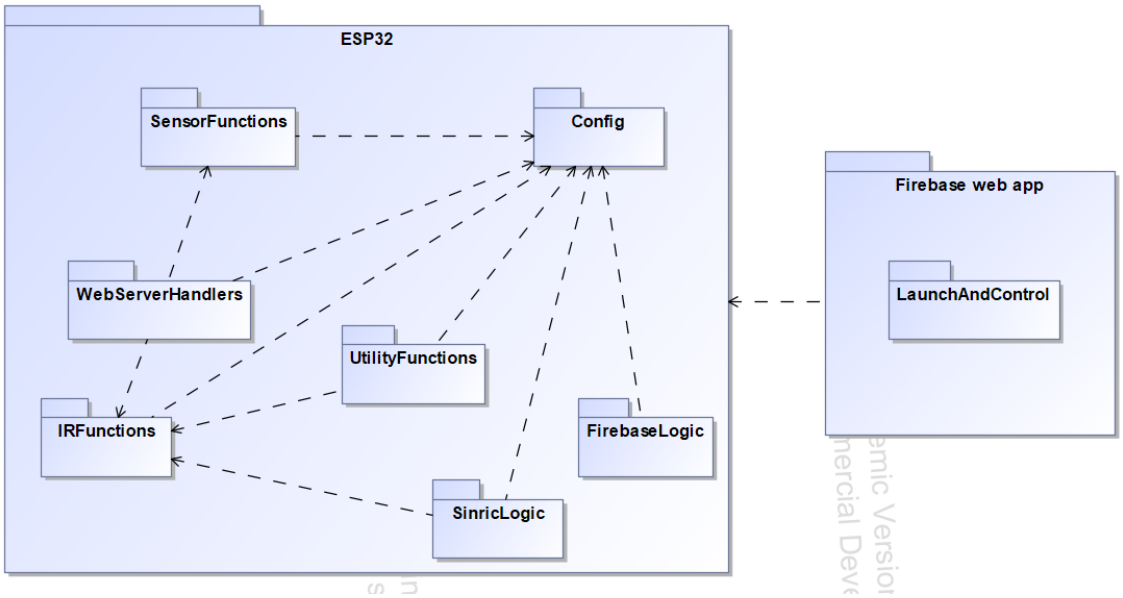
2.4.2. Prieigos reikalavimai

19 lentelė. NF15 specifikacija

Reikalavimas #:	**NF15 Reikalavimo tipas: V15.1 Įvykis/panaudojimo atvejis: Visi**		
Aprašymas:	Žiniatinklio programa turi būti apsaugota slaptažodžiu		
Pagrindimas:	Norima išvengti galimo įsilaužimo į sistemą.		
Šaltinis:	Sistemos užsakovas		
Tinkamumo kriterijus:	Žiniatinklio programa apsaugota slaptažodžiu		
Užsakovo tenkinimas:	4	Užsakovo netenkinimas:	4
Priklausomybės:	Nėra	Konfliktai:	Nėra
Papildoma medžiaga:			
Istorija:	Sukurtas: 2023-05-25		

2.5. Sistemos paketų diagrama

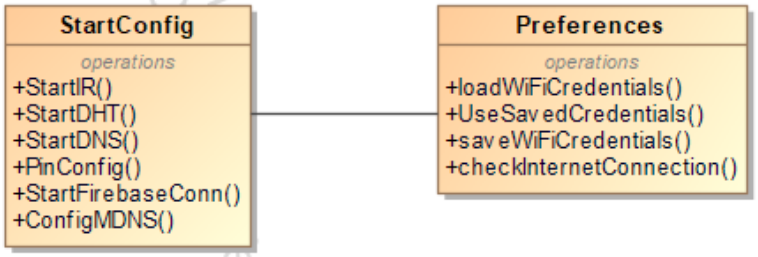
Pavaizduota sistemos paketų diagrama. Paketai atskirti ESP32 mikrovaldikliui ir išoriniai „Firebase“ žiniatinklio programai.



15 pav. Paketų diagrama

2.5.1. Detalizuotas „Config“ paketas

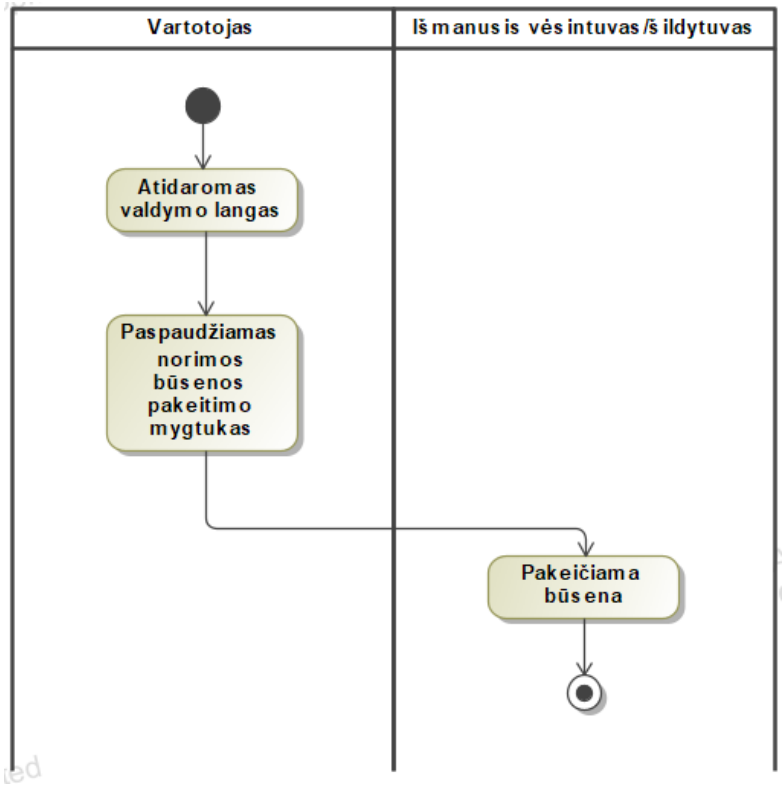
Šis paketas valdo mikrovaldiklio pradinį funkcijų paleidimą ir funkcijų konfigūracijas. Paketo „Config“ diagrama pateikta žemiau esančiame paveiksle.



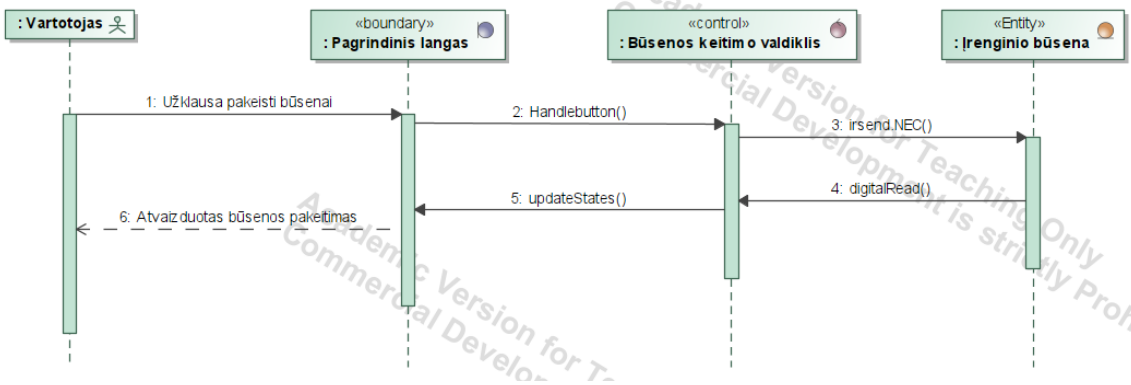
16 pav. „Config“ paketas

2.6. Sistemos dinaminis vaizdas

Žemiau pateika PA13 veiklos ir sąveikos diagramos.



17 pav. PA13 veiklos diagrama



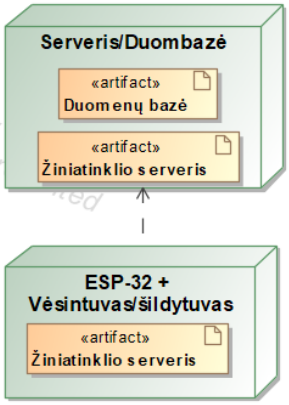
18 pav. PA13 sąveikos diagrama

2.7. Išdėstymo (angl. *deployment*) vaizdas

Sistema yra paskirstyta į du mazgus.

Šie mazgai yra:

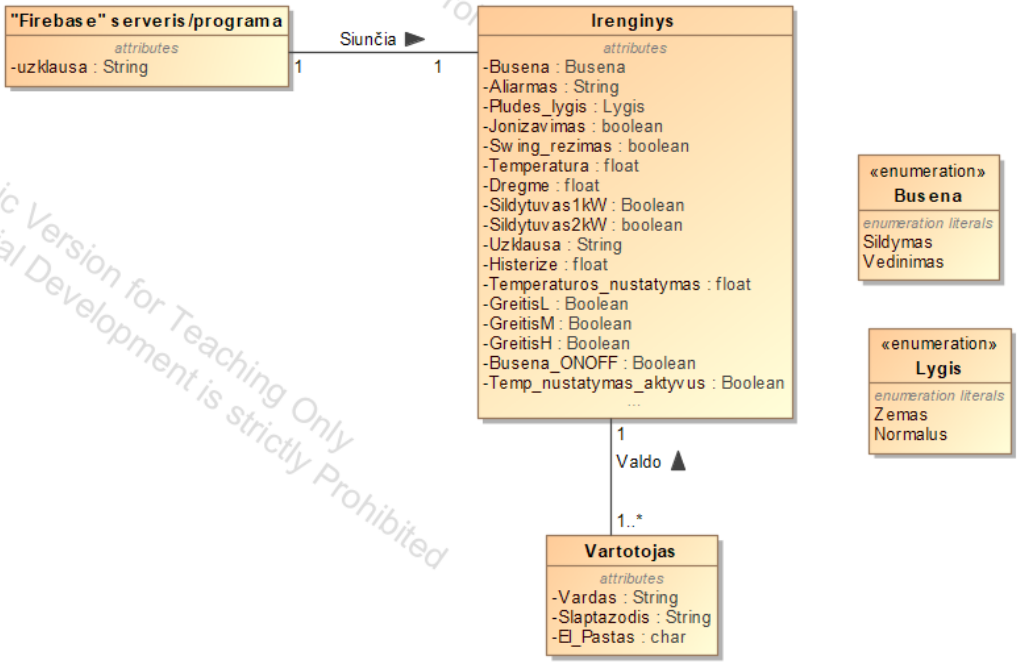
- **Serveris/duombazė:** nuotoliniam valdymui ir duomenų kaupimui.
- **Klientui duodamas išmanusis įrenginys:** „ESP32“ mikrovaldiklis ir vėsintuvas-šildytuvas.



19 pav. Išdėstymo diagrama

2.8. Duomenų vaizdas

Pateiktas paveikslas, kuriame pavaizduotas duomenų bazės modelis su visomis reikalingomis klasėmis.

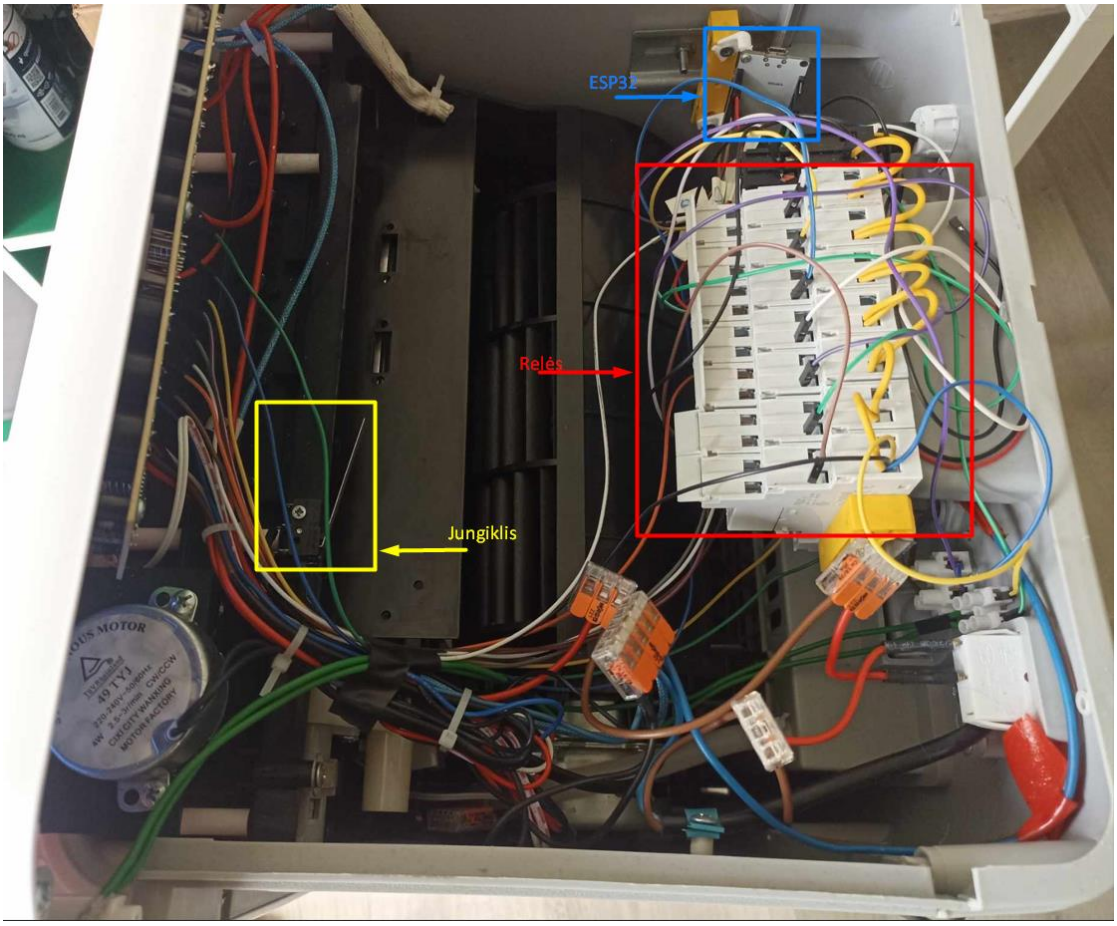


20 pav. Duomenų bazės modelis

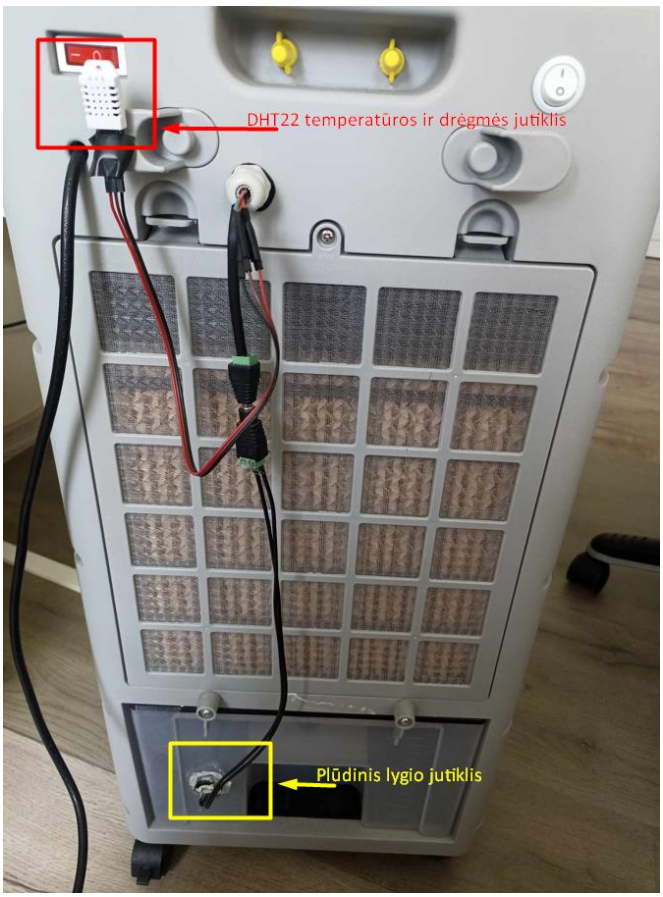
3. Tyrimo dalis

3.1. Įvadas

Magistrinio darbo metu buvo sukurtas išmanusis oro vėsintuvas-šildytuvas. Oro vėsintuvo-šildytuvo elektronikos dalys yra papildyta papildomu temperatūros ir drėgmės jutikliu, relėmis, dviem jungikliais, plūdiniu lygio jutikliu ir infraraudonųjų spindulių siųstuvu. Taip pat pridėtas „ESP32“ mikrovaldiklis. Šiame mikrovaldiklyje yra įkeltas kompiliuotas programos kodas. Papildomų apsaugų nuo atvirkštinės inžinerijos mikrovaldikliui nėra uždėta. Reikia atlikti „flash“ atminties šifravimą norint apsaugoti programos kodą.

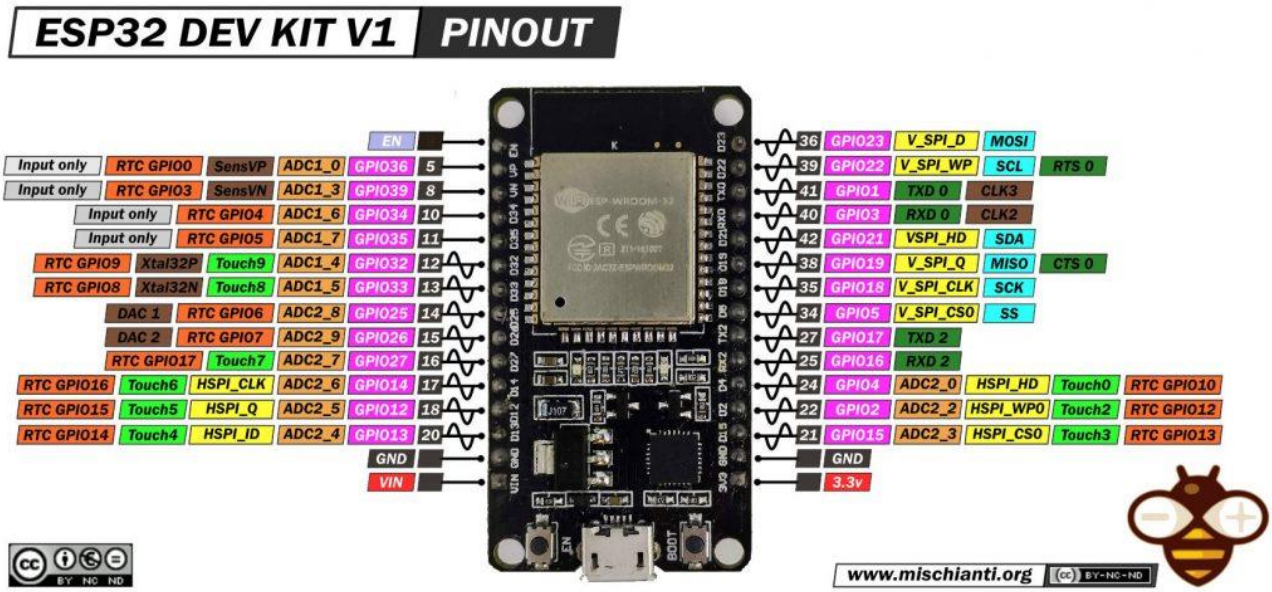


21 pav. Modifikuotas siurblio vidus



22 pav. Modifikuota siurblio išorė

Tyrimui bus naudojama tik viena išmaniojo oro vėsintuvo-šildytuvo dalis, tai „ESP32“ mikrovaldiklis.



23 pav. „ESP32 DOIT DEV KIT v1“ mikrovaldiklis [29]

3.2. Kokybės analizė

3.2.1. Statinė kodo analizė

Programų sistemos kodo statinei kodo analizei naudosime „Cppcheck“. Tai yra „C/C++“ kodo statinės analizės įrankis. Šis įrankis atlieka unikalią kodo analizę klaidoms aptikti ir daugiausia dėmesio skiria neapibrėžtam elgesiui ir pavojingoms kodavimo konstrukcijoms aptikti. Įrankis „Cppcheck“ sukurtas taip, kad galėtų analizuoti „C/C++“ kodą, net jei jis turi nestandartinę sintaksę, kuri pasitaiko įterptinėse sistemose [30].

Mikrovaldiklio programos kodas suskirstytas į „.ino“, „.cpp“ ir „.h“ failus. Failas su plėtiniu „.ino“ yra platformos „Arduino“ projekto failas. Failai su plėtiniu „.cpp“ yra „C++“ šaltinio failai. Failai su plėtiniu „.h“ yra „C++“ antraščių failai.

Name	Date modified	Type	Size
Config	1/24/2024 6:57 PM	CPP File	3 KB
config	1/24/2024 6:57 PM	C Header Source F...	1 KB
FirebaseLogic	1/25/2024 11:06 AM	CPP File	8 KB
FirebaseLogic	1/24/2024 7:59 PM	C Header Source F...	1 KB
Globals	1/24/2024 6:57 PM	C Header Source F...	3 KB
IRFunctions	1/25/2024 10:53 AM	CPP File	1 KB
IRFunctions	12/20/2023 9:45 PM	C Header Source F...	1 KB
Main	1/25/2024 10:53 AM	Arduino file	5 KB
SensorFunctions	1/22/2024 3:08 PM	CPP File	4 KB
SensorFunctions	1/16/2024 11:21 PM	C Header Source F...	1 KB
SinricLogic	1/23/2024 1:28 PM	CPP File	2 KB
SinricLogic	1/23/2024 12:26 AM	C Header Source F...	1 KB
UtilityFunctions	12/27/2023 9:49 PM	CPP File	1 KB
UtilityFunctions	12/27/2023 9:50 PM	C Header Source F...	1 KB
WebServerHandlers	1/24/2024 6:26 PM	CPP File	51 KB
WebServerHandlers	11/29/2023 10:00 AM	C Header Source F...	1 KB

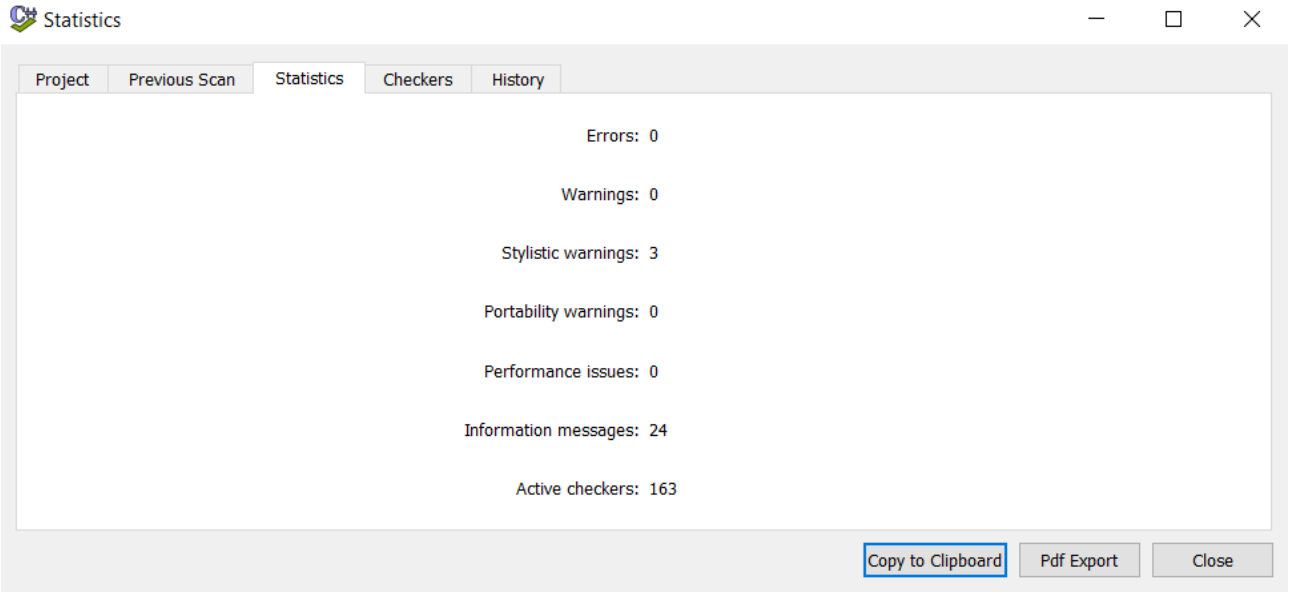
24 pav. Mikrovaldiklio programos failai.

Norinti atlikti statinę programos kodo analizę pirmiausia reikia atlikti modifikaciją „.ino“ failui. Failas „Main.ino“ turi būti paverstas į „.cpp“ failą, nes „Cppcheck“ neatpažįsta „Arduino“ platformos projekto failų. Pirmiausia reikia pridėti papildomą biblioteką „#include <Arduino.h>“ ir papildomą „int main()“ funkciją.

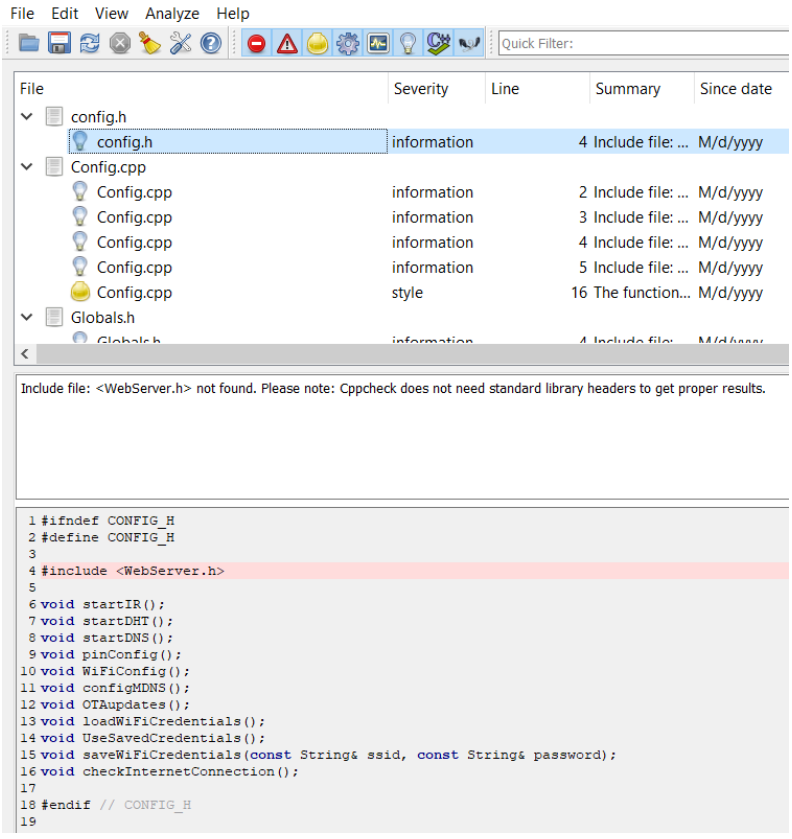
```
int main() {
    setup();
    while (true) {
        loop();
    }
    return 0;
}
```

25 pav. Papildoma „int main()“ funkcija

Atlikus analizę gauname 3 stiliaus įspėjimus ir 24 informacinius pranešimus. Visi informacinio lygiai pranešimai yra apie tai, kad „Cppcheck“ neranda nestandartinių bibliotekų. Nurodžius pilną kelią iki bibliotekos failų gauname daugiau klaidų. Šios bibliotekos yra pritaikytos „Arduino“ platformai, todėl į šiuos pranešimus nekreipiame dėmesio.

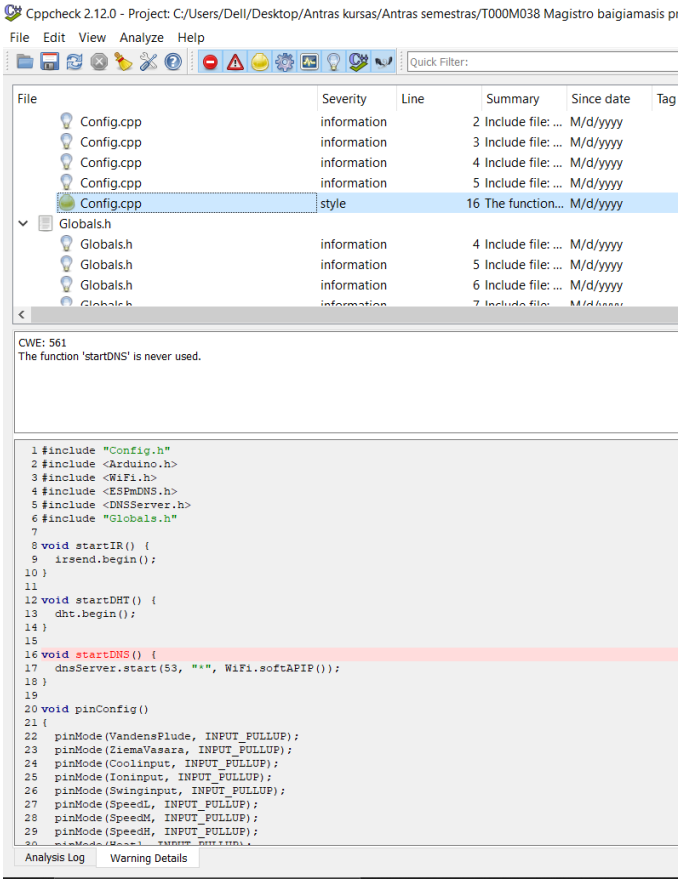


26 pav. Gauta statistika



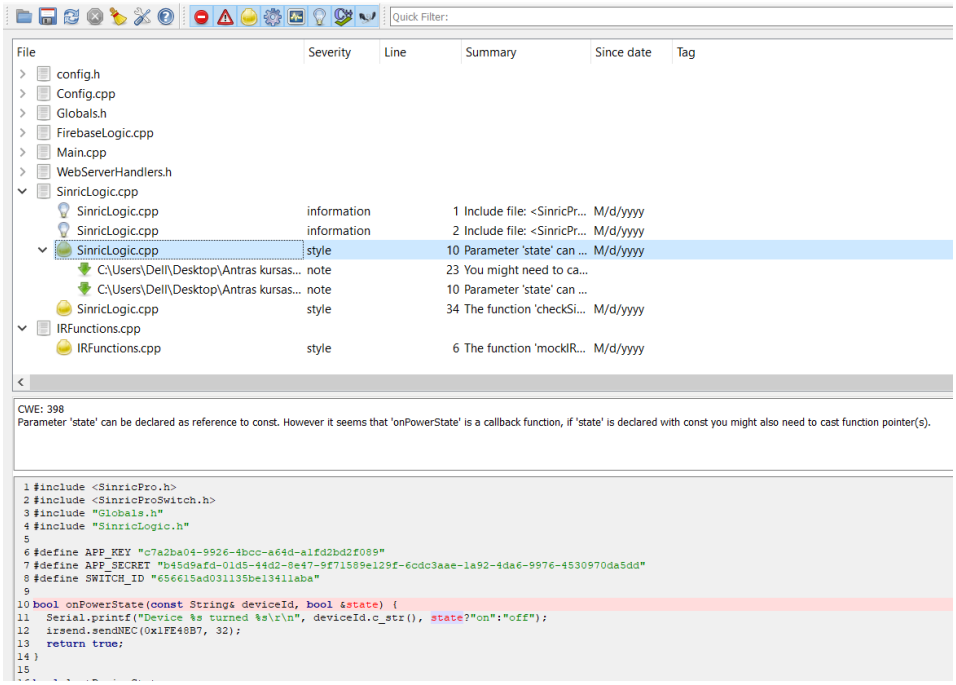
27 pav. Informacinis pranešimas dėl bibliotekos „config.h“ faile

Faile „config.cpp“ rasta niekur nenaudojama funkcija „startDNS()“. Šią funkciją galime pašalinti, taip palikdami daugiau vietos programos išplečiamumui.



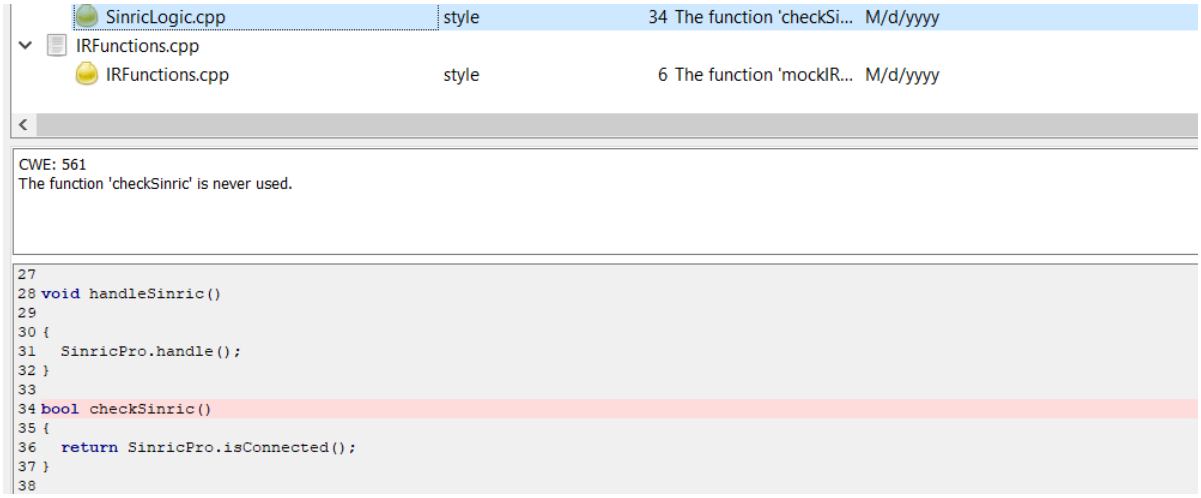
28 pav. Nenaudojama funkcija „config.cpp“ faile

Faile „SinricLogic.cpp“ randama rekomendacija, kad parametras „state“ gali būti deklaruojamas kaip „const“, siekiant nurodyti, kad jis neturėtų būti modifikuojamas naudojant funkciją „onPowerState“.



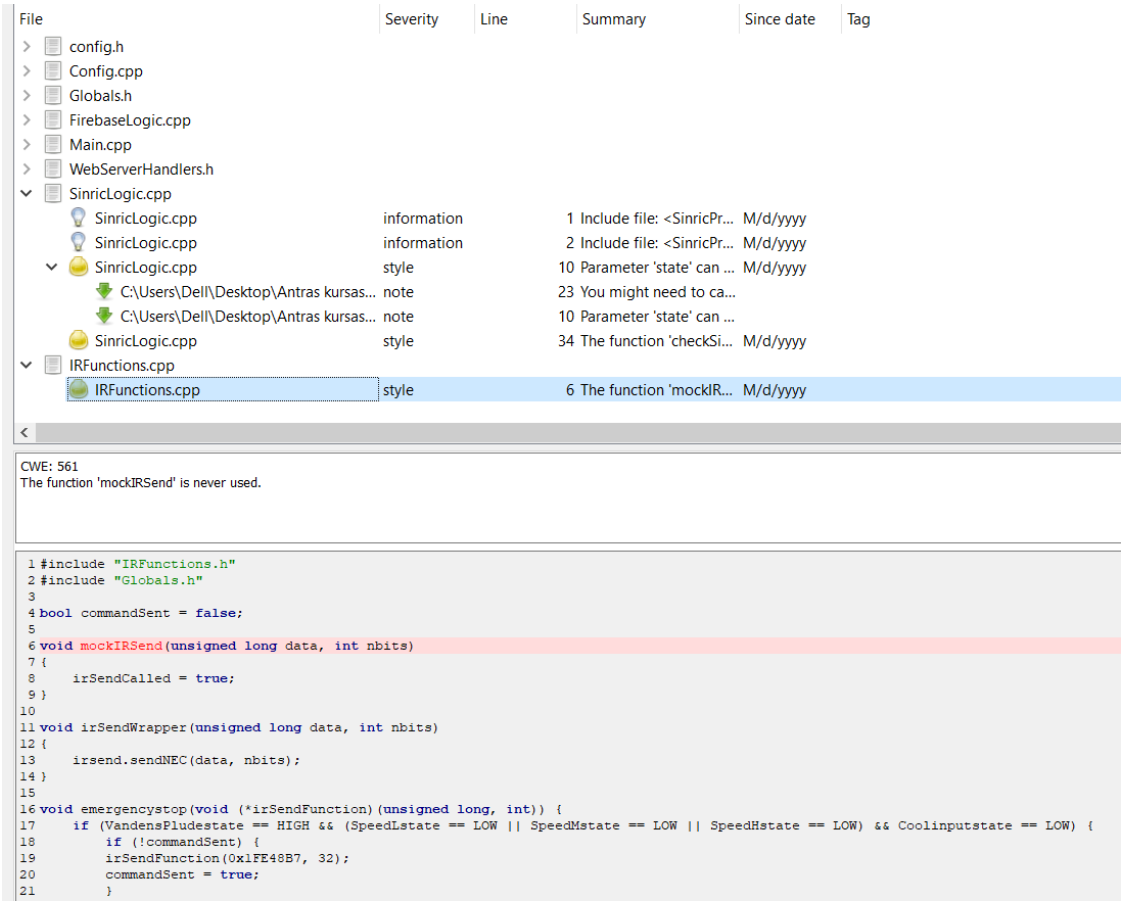
29 pav. Stiliaus klaida „SinricLogic.cpp“ faile

Taip pat dar randame nenaudojimą funkciją „checkSinric()“, kuria galime pašalinti, taip palikdami daugiau vietos programos išplečiamumui.



30 pav. Stiliaus klaida „SinricLogic.cpp“ faile

Faile „IRFunctions.cpp“ yra nenaudojama funkcija „mockIRSend“. Šią funkciją naudojame testavime. Testavimo modulis yra iškeltas į kitą vietą, todėl galime šią stiliaus klaidą ignoruoti arba iškelti funkciją į testavimo modulį.



31 pav. Stiliaus klaida „IRFunctions.cpp“ faile

Prieš pataisymus buvo sunaudota 1352597 baitai programos saugojimui.

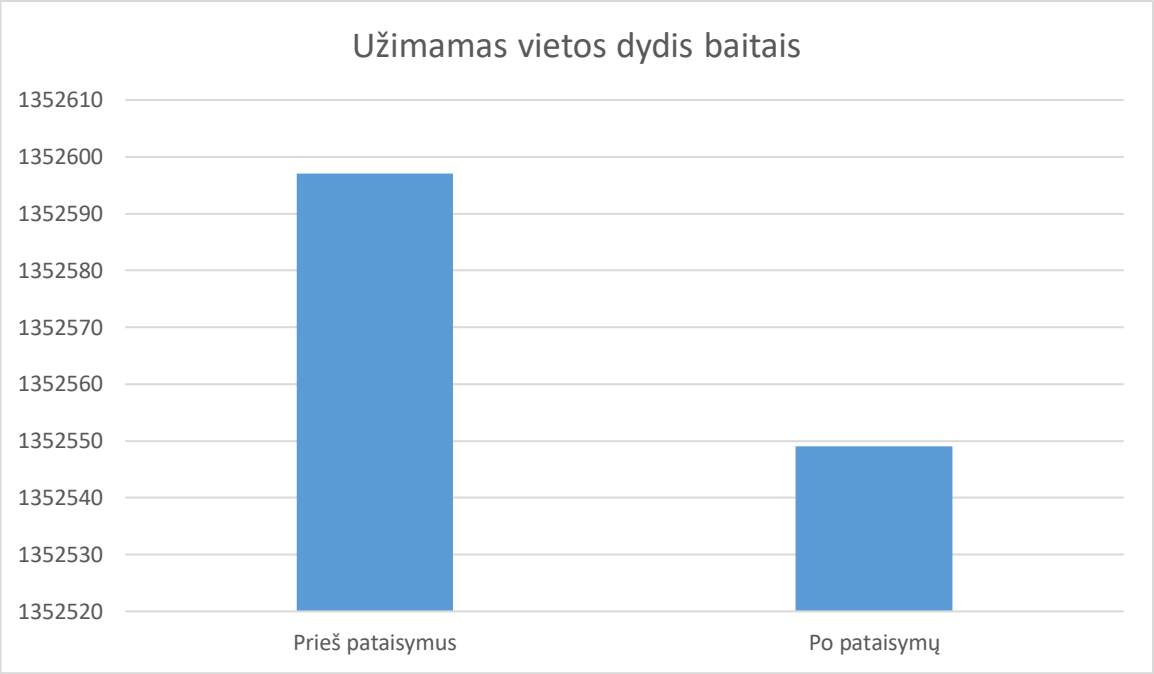
```
Sketch uses 1352597 bytes (64%) of program storage space. Maximum is 2097152 bytes.
Global variables use 51732 bytes (15%) of dynamic memory, leaving 275948 bytes for local variables. Maximum is 327680 bytes.
```

32 pav. Užimama vieta prieš pataisymus

Po pataisymų buvo sunaudota 1352549 baitų programos saugojimui.

```
Sketch uses 1352549 bytes (64%) of program storage space. Maximum is 2097152 bytes.
Global variables use 51732 bytes (15%) of dynamic memory, leaving 275948 bytes for local variables. Maximum is 327680 bytes.
```

33 pav. Užimama vieta po pataisymų



34 pav. Užimama programos vieta

Užimama programos vieta sumažėjo 48 baitais.

4. Eksperimentinė dalis

4.1. Tyrimo hipotezė

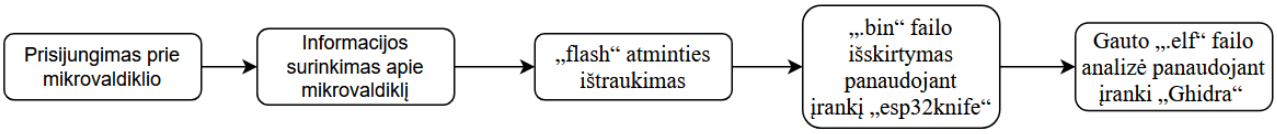
H1: Turėtų sumažėti randamos naudingos informacijos kiekis apie programą panaudojus „flash“ atminties šifravimą.

4.2. Hipotezės patikrinimas

Hipotezės tikrinimas bus vykdomas tikrinant randamos informacijos kiekį apie programą neapsaugojus mikrovaldiklio atminties turinio ir apsaugojus mikrovaldiklio atmintį panaudojus „flash“ atminties šifravimą.

4.3. Neapsaugoto įrenginio programos kodo pasisavinimo procesas

Neapsaugoto mikrovaldiklio programos kodo pasisavinimo procesą sudaro šeši etapai, kurie yra atvaizduoti srauto diagramoje.



35 pav. „ESP32“ mikrovaldiklio informacijos išgavimo procesas

4.3.1. ESP32 "flash" atminties turinio išgavimas

„Esptool“ yra atvirojo kodo „Python“ biblioteka, skirta sąveikai su „Espressif“ mikrovaldikliais, ypač „ESP8266“ ir „ESP32“ [31].

Atsisiuntus ir įsidiegus įrankį galime pradėti darbą. Pirmiausia prisijungiame „ESP32“ mikrovaldiklį prie kompiuterio naudodami USB kabelį. Atliekame mikrovaldiklio tapatybės nustatymą panaudodami komandą „python -m esptool flash_id“.

```
C:\Users\Dell>python -m esptool flash_id
esptool.py v4.7.0
Found 1 serial ports
Serial port COM3
Connecting.....
Detecting chip type... Unsupported detection protocol, switching and trying again...
Connecting....
Detecting chip type... ESP32
Chip is ESP32-D0WD-V3 (revision v3.1)
Features: WiFi, BT, Dual Core, 240MHz, VRef calibration in efuse, Coding Scheme None
Crystal is 40MHz
MAC: 48:e7:29:9e:c6:44
Uploading stub...
Running stub...
Stub running...
Manufacturer: 54
Device: 4016
Detected flash size: 4MB
Hard resetting via RTS pin...
```

36 pav. „ESP32“ mikrovaldiklio informacija

- **Naudojama esptool.py versija:** 4.7.0
- **Aptikti serijiniai prievadai:** COM3
- **Mikrokontrolerio tipas:** „ESP32-DOWD-V3” (revizija v3.1)
- **Mikrokontrolerio funkcijos:** „WiFi”, „BT”, dviejų branduolių, 240 MHz, VRef kalibravimas „efuse”, kodavimo schemos nėra.
- **Kristalo dažnis:** 40 MHz
- **MAC adresas:** 48:27:29:9e:c6:44
- **Gamintojas:** 54
- **Įrenginys:** 4016
- **Aptikta flash atminties talpa:** 4 MB

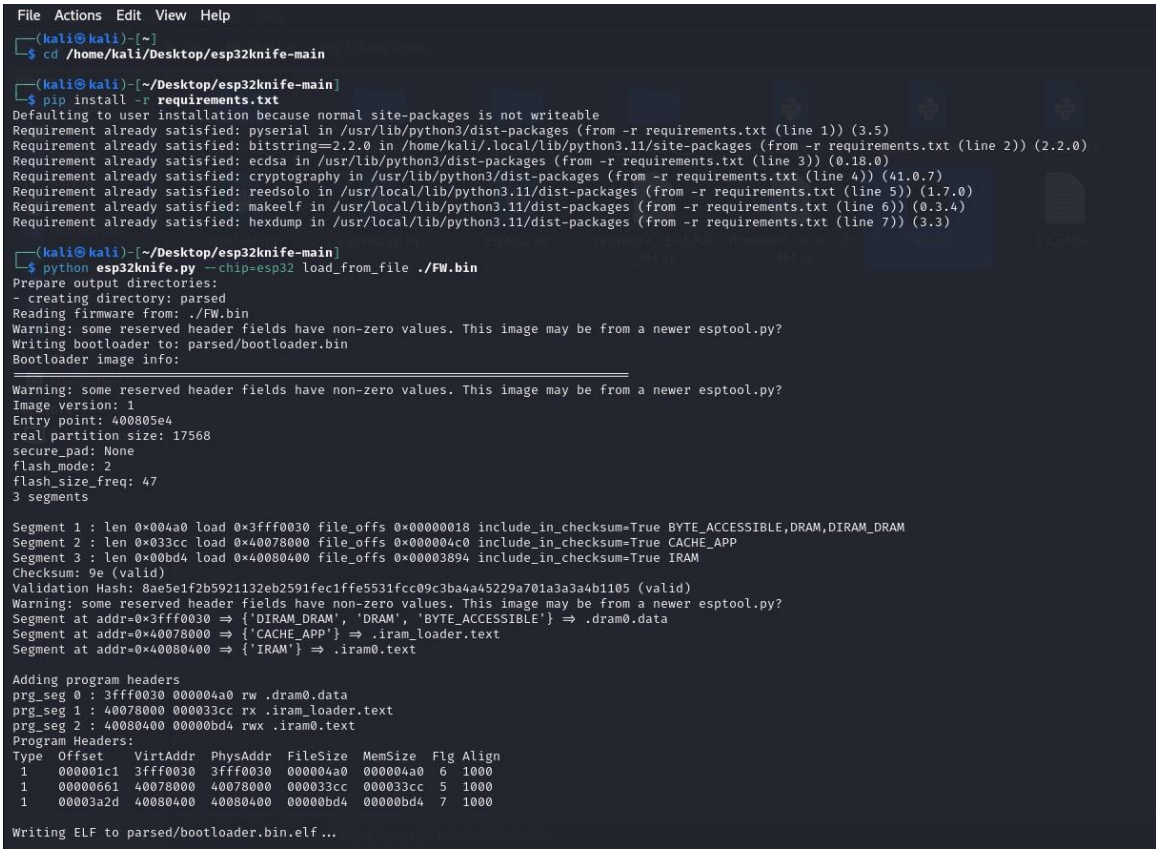
Kelis iš šių gautų duomenų panaudosime kitai reikalingai komandai su kuria išgausime mikrovaldiklio atminties turinį. Turinio ištraukimui naudojame šią komandą: „python -m esptool --baud 115200 --port COM3 read_flash 0x0 0x400000 FW.bin“

- **python -m esptool:** tai paleidžia „esptool“ kaip modulį iš „Python“ aplinkos. Reikia turėti įdiegtą „Python“ aplinką ir „esptool“ biblioteką.
- **--baud 115200:** nustatoma duomenų perdavimo sparta, naudojamą ryšiui su įrenginiu. Čia nurodytas greitis yra 115200 bitų per sekunde.
- **--port COM3:** nurodo, kuris kompiuterio prievadas bus naudojamas ryšiui su „ESP“ įrenginiu. Šiuo atveju tai yra „COM3“ prievadas, dažniausiai naudojamas „Microsoft Windows“ sistemose.
- **read_flash:** ši komanda nurodo, kad reikia nuskaityti atmintį iš „ESP“ įrenginio.
- **0x0 0x400000:** nurodo atminties bloko pradžios ir pabaigos adresus. Čia skaitymas pradedamas nuo adreso 0x0 ir tęsiamas iki 0x400000 (4 megabaitai).
- **FW.bin:** nurodo failo vardą, į kurį bus įrašyti atmintyje esantys duomenys.

```
C:\Users\Dell>python -m esptool --baud 115200 --port COM3 read_flash 0x0 0x400000 FW.bin
esptool.py v4.7.0
Serial port COM3
Connecting....
Detecting chip type... Unsupported detection protocol, switching and trying again...
Connecting.......
Detecting chip type... ESP32
Chip is ESP32-D0WD-V3 (revision v3.1)
Features: WiFi, BT, Dual Core, 240MHz, VRef calibration in efuse, Coding Scheme None
Crystal is 40MHz
MAC: 48:e7:29:9e:c6:44
Uploading stub...
Running stub...
Stub running...
4194304 (100 %)
4194304 (100 %)
Read 4194304 bytes at 0x00000000 in 381.9 seconds (87.9 kbit/s)...
Hard resetting via RTS pin...
```

37 pav. „ESP32“ atminties turinio ištraukimas į „.bin“ failą

Atlikus šiuos veiksmus gauname „FW.bin“ failą. Šio failą struktūrą panagrinėjame panaudojant įrankį „esp32knife“. Taip pat šis failas sukuria „ELF“ failą, kuri galima panaudoti analizei naudojant įrankį „Ghidra“.



38 pav. Įrankio „esp32knife“ panaudojimas

4.3.2. Gauto atminties turinio analizė

Panaudojus įrankį „esp32knife“ gauname atminties skirsnių lentelę.

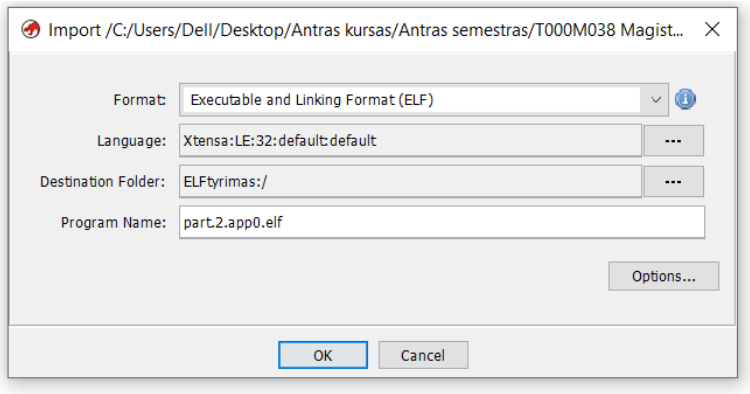
# ESP-IDF Partition Table					
# Name	Type	SubType	Offset	Size	Flags
nvs	data	nvs	0x9000	20K	
otadata	data	ota	0xe000	8K	
app0	app	ota_0	0x10000	2M	
spiffs	data	spiffs	0x210000	1920K	
coredump	data	coredump	0x3f0000	64K	

39 pav. „ESP32“ skirsnių lentelė

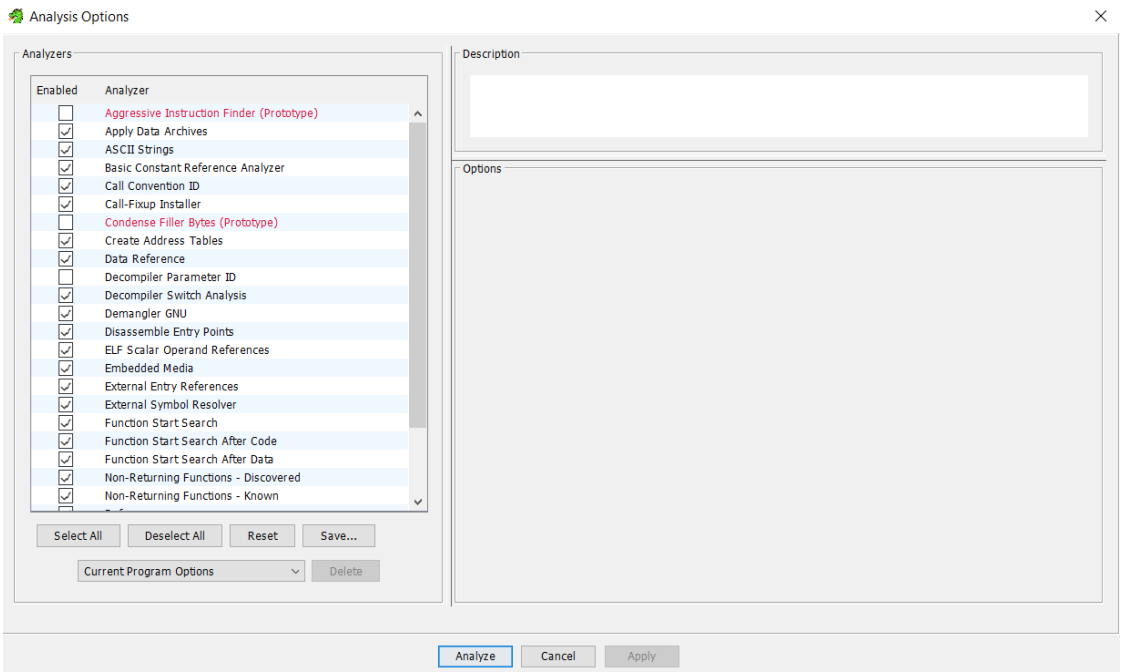
– **„nvs“:** šiame skirsnyje saugomi nekintamieji duomenys, pavyzdžiui, „WiFi“ duomenys, įrenginio „PHY“ kalibravimo duomenys ir bet kokie kiti duomenys, kurie turi būti saugomi nekintamojoje atmintyje [32].

– **„otadata“:** šis skirsnis naudojamas duomenims, skirtiems „OTA (over-the-air)“ atnaujinimams, saugoti. Šis skirsnis naudojamas tik tada, kai „OTA“ yra naudojamas projekte. Šio skirsnio dydis turėtų būti fiksuotas 8 kilobaitai [32].
– **„app0“:** šiame skirsnyje saugomas pagrindinis programos kodas [32].
– **„spiffs“:** šis skirsnis apibrėžia „SPI flash“ failų sistemos naudojimą, jis taip pat tinka didesniems failams ir atlieka atminties nusidėvėjimo išlyginimą bei failų sistemos nuoseklumo patikrą [32].
– **„coredump“:** branduolio išrašo skirsnio potipis naudojamas branduolio išrašo saugojimui „flash“ atmintinėje. Branduolio išrašas naudojamas kritinėms klaidoms, tokioms kaip avarija ir panika, analizuoti [32].

Tolimesnė analizė atliekama naudojant įrankį „Ghidra“. Analizei naudojamas gautas „part.2.app0.elf“ failas.

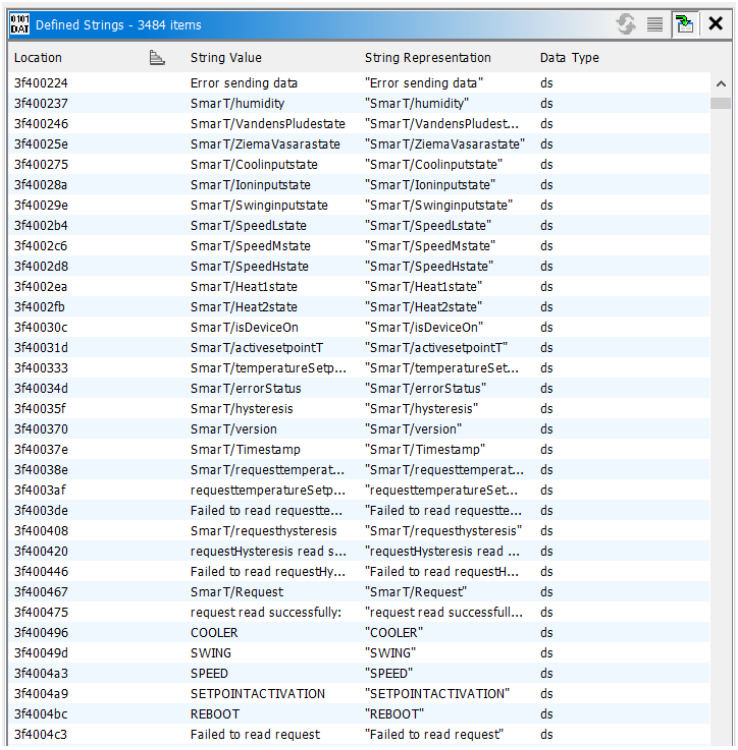


40 pav. Analizuojamo sistemos failo konfigūracijos pasirinkimas



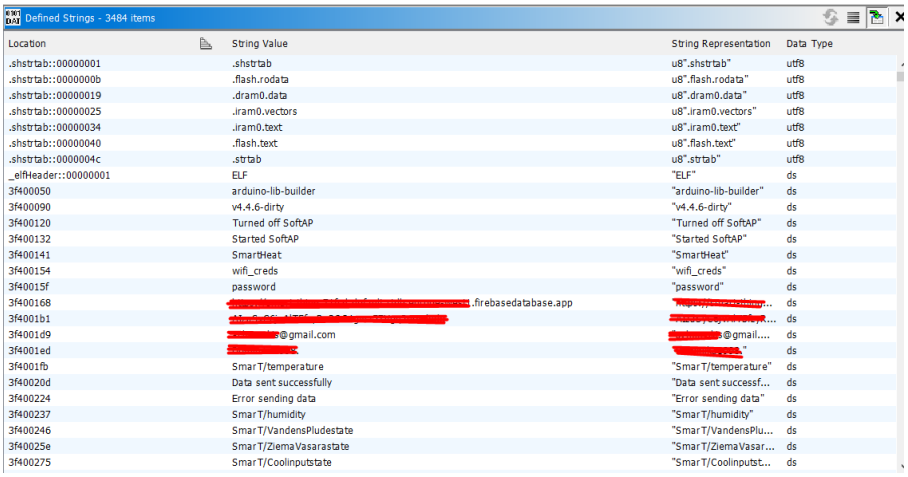
41 pav. Analizavimo pasirinktys

Atlikus analizuojamo sistemos failo analizę randame 3484 simbolių eilutes. Šiuose eilutėse yra naudingos informacijos, tokios kaip sistemos būsenos, duomenų bazės keliai, HTML žymėjimo kalbos elementai, naudojamų kodo elementų versijos, prisijungimo duomenys.


Defined Strings - 3484 items

Location	String Value	String Representation	Data Type
3f400224	Error sending data	"Error sending data"	ds
3f400237	SmarT/humidity	"SmarT/humidity"	ds
3f400246	SmarT/VandensPludestate	"SmarT/VandensPludest...	ds
3f40025e	SmarT/ZiemaVasarastate	"SmarT/ZiemaVasarastate"	ds
3f400275	SmarT/Coolinputstate	"SmarT/Coolinputstate"	ds
3f40028a	SmarT/Ioninputstate	"SmarT/Ioninputstate"	ds
3f40029e	SmarT/Swinginputstate	"SmarT/Swinginputstate"	ds
3f4002b4	SmarT/SpeedLstate	"SmarT/SpeedLstate"	ds
3f4002c6	SmarT/SpeedMstate	"SmarT/SpeedMstate"	ds
3f4002d8	SmarT/SpeedHstate	"SmarT/SpeedHstate"	ds
3f4002ea	SmarT/Heat1state	"SmarT/Heat1state"	ds
3f4002fb	SmarT/Heat2state	"SmarT/Heat2state"	ds
3f40030c	SmarT/isDeviceOn	"SmarT/isDeviceOn"	ds
3f40031d	SmarT/activesetpointT	"SmarT/activesetpointT"	ds
3f400333	SmarT/temperatureSetp...	"SmarT/temperatureSet...	ds
3f40034d	SmarT/errorStatus	"SmarT/errorStatus"	ds
3f40035f	SmarT/hysteresis	"SmarT/hysteresis"	ds
3f400370	SmarT/version	"SmarT/version"	ds
3f40037e	SmarT/Timestamp	"SmarT/Timestamp"	ds
3f40038e	SmarT/requesttemperat...	"SmarT/requesttemperat...	ds
3f4003af	requesttemperatureSetp...	"requesttemperatureSet...	ds
3f4003de	Failed to read requestte...	"Failed to read requestte...	ds
3f400408	SmarT/requesthysteresis	"SmarT/requesthysteresis"	ds
3f400420	requestHysteresis read s...	"requestHysteresis read ...	ds
3f400446	Failed to read requestHy...	"Failed to read requestH...	ds
3f400467	SmarT/Request	"SmarT/Request"	ds
3f400475	request read successfully:	"request read successfull...	ds
3f400496	COOLER	"COOLER"	ds
3f40049d	SWING	"SWING"	ds
3f4004a3	SPEED	"SPEED"	ds
3f4004a9	SETPOINTACTIVATION	"SETPOINTACTIVATION"	ds
3f4004bc	REBOOT	"REBOOT"	ds
3f4004c3	Failed to read request	"Failed to read request"	ds

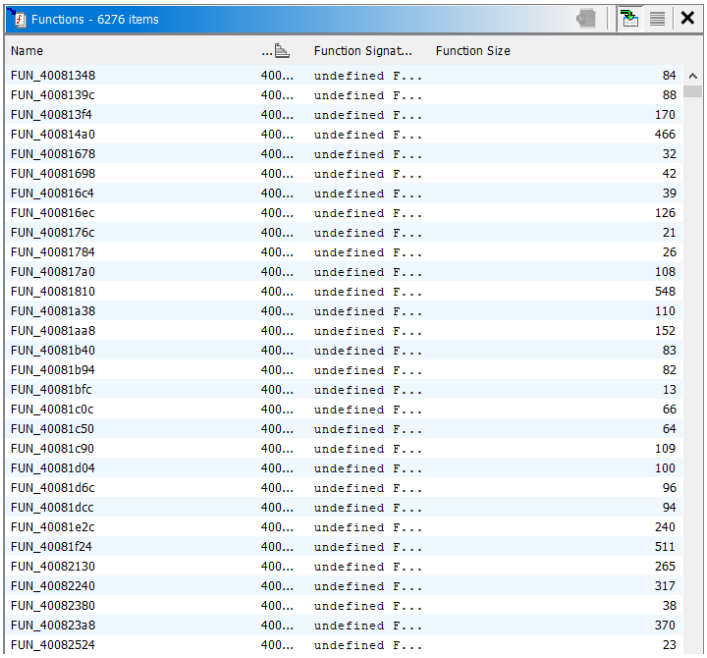
42 pav. Rastos simbolių eilutės


Defined Strings - 3484 items

Location	String Value	String Representation	Data Type
.shstrtab::00000001	.shstrtab	u8".shstrtab"	utf8
.shstrtab::0000000b	.flash.rodata	u8".flash.rodata"	utf8
.shstrtab::00000019	.dram0.data	u8".dram0.data"	utf8
.shstrtab::00000025	.iram0.vectors	u8".iram0.vectors"	utf8
.shstrtab::00000034	.iram0.text	u8".iram0.text"	utf8
.shstrtab::00000040	.flash.text	u8".flash.text"	utf8
.shstrtab::0000004c	.strtab	u8".strtab"	utf8
_elfHeader::00000001	ELF	"ELF"	ds
3f400050	arduino-lib-builder	"arduino-lib-builder"	ds
3f400090	v4.4.6-dirty	"v4.4.6-dirty"	ds
3f400120	Turned off SoftAP	"Turned off SoftAP"	ds
3f400132	Started SoftAP	"Started SoftAP"	ds
3f400141	SmartHeat	"SmartHeat"	ds
3f400154	wifi_creds	"wifi_creds"	ds
3f40015f	password	"password"	ds
3f400168	[redacted].firebasedatabase.app	[redacted]	ds
3f4001b1	[redacted]	[redacted]	ds
3f4001d9	[redacted]@gmail.com	[redacted]@gmail....	ds
3f4001ed	[redacted]	[redacted]	ds
3f4001fb	SmarT/temperature	"SmarT/temperature"	ds
3f40020d	Data sent successfully	"Data sent successf...	ds
3f400224	Error sending data	"Error sending data"	ds
3f400237	SmarT/humidity	"SmarT/humidity"	ds
3f400246	SmarT/VandensPludestate	"SmarT/VandensPlu...	ds
3f40025e	SmarT/ZiemaVasarastate	"SmarT/ZiemaVasar...	ds
3f400275	SmarT/Coolinputstate	"SmarT/Coolinputst...	ds

43 pav. Rastos simbolių eilutės

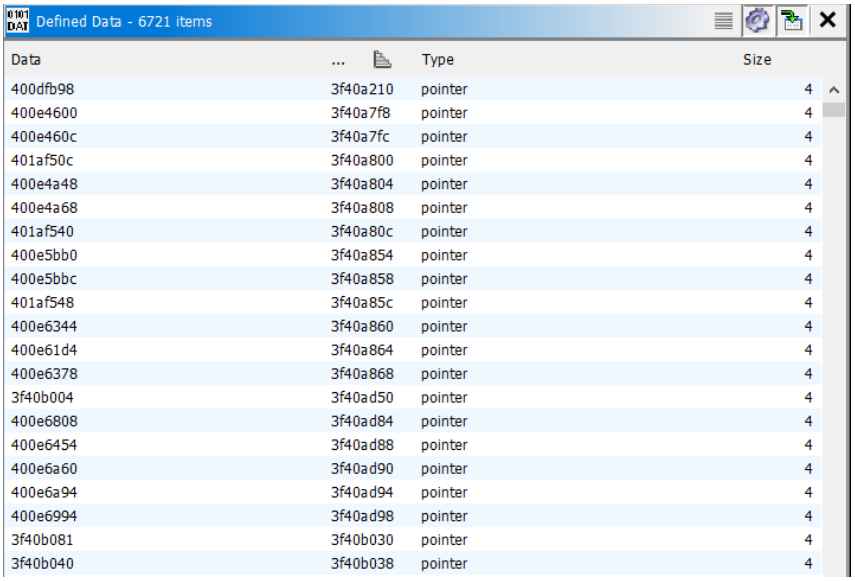
Atlikus analizuojamo sistemos failo analizę randame 6276 funkcijas. Šių funkcijų skaičius yra toks dėl mikrovaldiklio, pagrindinio programų sistemos kodo ir bibliotekų funkcijų bendros sumos.


Functions - 6276 items

Name	...	Function Signat...	Function Size
FUN_40081348	400...	undefined F...	84
FUN_4008139c	400...	undefined F...	88
FUN_400813f4	400...	undefined F...	170
FUN_400814a0	400...	undefined F...	466
FUN_40081678	400...	undefined F...	32
FUN_40081698	400...	undefined F...	42
FUN_400816c4	400...	undefined F...	39
FUN_400816ec	400...	undefined F...	126
FUN_4008176c	400...	undefined F...	21
FUN_40081784	400...	undefined F...	26
FUN_400817a0	400...	undefined F...	108
FUN_40081810	400...	undefined F...	548
FUN_40081a38	400...	undefined F...	110
FUN_40081aa8	400...	undefined F...	152
FUN_40081b40	400...	undefined F...	83
FUN_40081b94	400...	undefined F...	82
FUN_40081bfc	400...	undefined F...	13
FUN_40081c0c	400...	undefined F...	66
FUN_40081c50	400...	undefined F...	64
FUN_40081c90	400...	undefined F...	109
FUN_40081d04	400...	undefined F...	100
FUN_40081d6c	400...	undefined F...	96
FUN_40081dcc	400...	undefined F...	94
FUN_40081e2c	400...	undefined F...	240
FUN_40081f24	400...	undefined F...	511
FUN_40082130	400...	undefined F...	265
FUN_40082240	400...	undefined F...	317
FUN_40082380	400...	undefined F...	38
FUN_400823a8	400...	undefined F...	370
FUN_40082524	400...	undefined F...	23

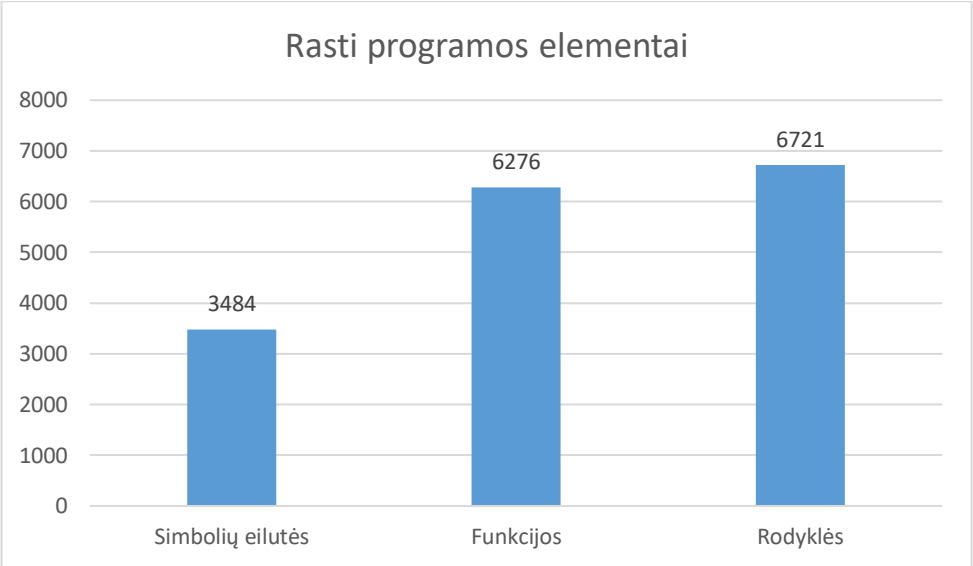
44 pav. Rastos funkcijos

Atlikus analizuojamo sistemos failo analizę randame 6721 rodykles.


Defined Data - 6721 items

Data	...	Type	Size
400dfb98	3f40a210	pointer	4
400e4600	3f40a7f8	pointer	4
400e460c	3f40a7fc	pointer	4
401af50c	3f40a800	pointer	4
400e4a48	3f40a804	pointer	4
400e4a68	3f40a808	pointer	4
401af540	3f40a80c	pointer	4
400e5bb0	3f40a854	pointer	4
400e5bbc	3f40a858	pointer	4
401af548	3f40a85c	pointer	4
400e6344	3f40a860	pointer	4
400e61d4	3f40a864	pointer	4
400e6378	3f40a868	pointer	4
3f40b004	3f40ad50	pointer	4
400e6808	3f40ad84	pointer	4
400e6454	3f40ad88	pointer	4
400e6a60	3f40ad90	pointer	4
400e6a94	3f40ad94	pointer	4
400e6994	3f40ad98	pointer	4
3f40b081	3f40b030	pointer	4
3f40b040	3f40b038	pointer	4

45 pav. Rastos rodyklės



46 pav. Rasti programos elementai panaudojant „Ghidra“

Randame ir dekompiliuotų funkcijų, kurios yra žmogui suprantamos ir panašios į sukurtos programų sistemos kodo funkcijas. Viena iš jų į avarinių pranešimų formavimo funkciją. Ši funkcija formuoja avarinius pranešimus pagal tam tikras sąlygas. Tai matome ir dekompiliuotame kode.

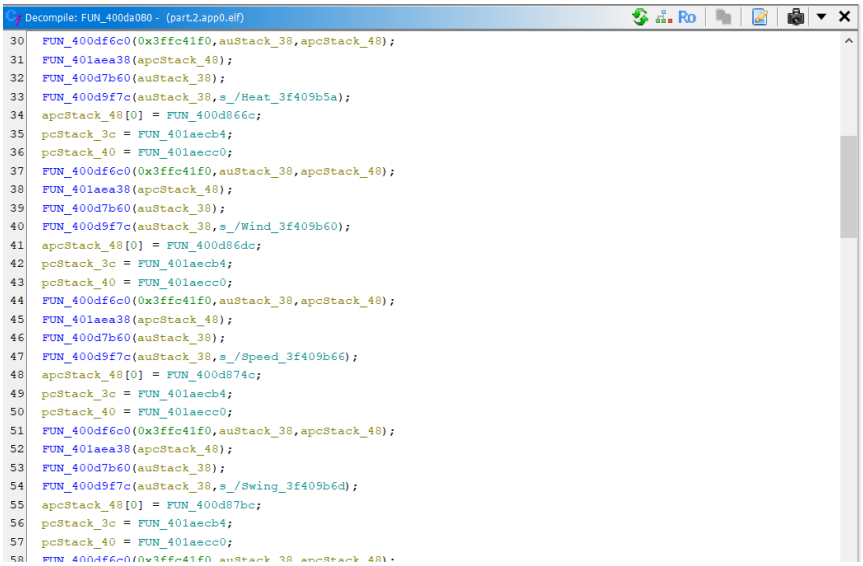
```
Decompile: FUN_400d397c - (part.2.app0.elf)
4  void FUN_400d397c(void)
5
6  {
7    undefined uVar1;
8    int iVar2;
9    uint uVar3;
10   undefined auStack_34 [16];
11   int iStack_24;
12
13   memw();
14   memw();
15   iStack_24 = _DAT_3ffc4e64;
16   FUN_40109f04(0x3ffc436c,&DAT_3f426903);
17   if (((_DAT_3ffc4368 < -20.0) || (50.0 < _DAT_3ffc4368)) || (NAN(_DAT_3ffc4368))) {
18     FUN_40109d84(auStack_34,s_Temperature_sensor_out_of_range;_3f400520);
19     iVar2 = FUN_4010a574(0x3ffc436c,auStack_34);
20     FUN_40109bdc(auStack_34);
21     if (iVar2 == -1) {
22       FUN_4010a15c(0x3ffc436c,s_Temperature_sensor_out_of_range;_3f400520);
23     }
24   }
25   if (((_DAT_3ffc4364 < 0.0) || (100.0 < _DAT_3ffc4364)) || (NAN(_DAT_3ffc4364))) {
26     FUN_40109d84(auStack_34,s_Humidity_sensor_out_of_range_or_n_3f400542);
27     iVar2 = FUN_4010a574(0x3ffc436c,auStack_34);
28     FUN_40109bdc(auStack_34);
29     if (iVar2 == -1) {
30       FUN_4010a15c(0x3ffc436c,s_Humidity_sensor_out_of_range_or_n_3f400542);
31     }
32   }
33   iVar2 = FUN_400dbeec();
34   if (iVar2 != 3) {
35     FUN_40109d84(auStack_34,s_WiFi_not_connected;_3f400573);
36     iVar2 = FUN_4010a574(0x3ffc436c,auStack_34);
37     FUN_40109bdc(auStack_34);
```

47 pav. Avarinių pranešimų formavimo dekompiliuotos funkcijos fragmentas

```
void checkForErrors() {
  errorStatus = "";
  if ((temperatura < -20 || temperatura > 50 || isnan(temperatura)) &&
      errorStatus.indexOf("Temperature sensor out of range; ") == -1) {
    errorStatus += "Temperature sensor out of range; ";
  }
  if ((dregme < 0 || dregme > 100 || isnan(dregme)) &&
      errorStatus.indexOf("Humidity sensor out of range or not responding; ") == -1) {
    errorStatus += "Humidity sensor out of range or not responding; ";
  }
  if ((WiFi.status() != WL_CONNECTED) &&
      errorStatus.indexOf("WiFi not connected; ") == -1) {
    errorStatus += "WiFi not connected; ";
  }
```

48 pav. Avarinių pranešimų formavimo funkcijos fragmentas

Kita rasta funkcija yra skirta žiniatinklio serverio nustatymams ir paleidimui. Paveiksluose pavaizduoti įvykių klausytojų priskyrimas sistemos funkcijoms. Tai matome dekompiliuotame kode ir programų sistemos kode.

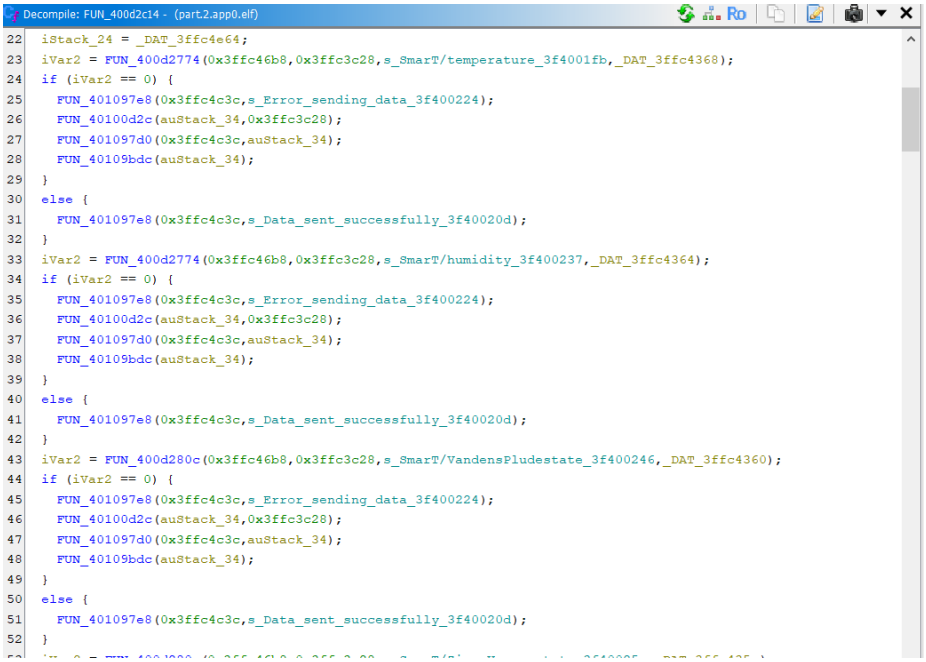


49 pav. Serverio paleidimo dekompiliuotos funkcijos fragmentas

```
server.on("/Heat", handleButton2);
server.on("/Wind", handleButton3);
server.on("/Speed", handleButton4);
server.on("/Swing", handleButton5);
server.on("/Cooler", handleButton6);
```

50 pav. Serverio paleidimo funkcijos fragmentas

Kita rasta funkcija yra skirta duomenų siuntimui į „Firebase“ duomenų bazę. Tai matome ir dekompiliuotame ir programų sistemų kode.



51 pav. Duomenų siuntimo į duomenų bazę dekompliuotos funkcijos fragmentas

```
void sendDataToRDBS() {
    if (Firebase.RTDB.setFloat(&fbdo, "SmarT/temperature", temperatura)) {
      Serial.println("Data sent successfully");
    } else {
      Serial.println("Error sending data");
      Serial.println(fbdo.errorReason());
    }
    if (Firebase.RTDB.setFloat(&fbdo, "SmarT/humidity", dregme)) {
      Serial.println("Data sent successfully");
    } else {
      Serial.println("Error sending data");
      Serial.println(fbdo.errorReason());
    }
    if (Firebase.RTDB.setInt(&fbdo, "SmarT/VandensPludestate", VandensPludestate)) {
      Serial.println("Data sent successfully");
    } else {
      Serial.println("Error sending data");
      Serial.println(fbdo.errorReason());
    }
```

52 pav. Duomenų siuntimo į duomenų bazę funkcijos fragmentas

Kita rasta funkcija yra skirta mikrovaldiklio prisijungimo taško valdymui. Prisijungimo valdymo taško valdymas vyksta pagal sąlygas ir tai matome dekompiliuotame ir programų sistemų kode.

Decompile: FUN_400d2254 - (part.2.app0.elf)

```
1
2 void FUN_400d2254(void)
3
4 {
5   int iVar1;
6   char *pcVar2;
7
8   FUN_40107b14(5000);
9   iVar1 = FUN_400dbeec();
10  if ((iVar1 == 3) && (DAT_3ffc430d != '\0')) {
11    FUN_400db738(1);
12    FUN_400da798(0x3ffc4634,1);
13    pcVar2 = s_Turned_off_SoftAP_3f400120;
14    DAT_3ffc430d = 0;
15  }
16  else {
17    iVar1 = FUN_400dbeec();
18    if (iVar1 == 3) {
19      return;
20    }
21    if (DAT_3ffc430d != '\0') {
22      return;
23    }
24    FUN_400db738(3);
25    FUN_400da68c(0x3ffc4634,PTR_s_SmartHeatAP_3ffbdb6c,0,1,0,4,0);
26    DAT_3ffc430d = 1;
27    pcVar2 = s_Started_SoftAP_3f400132;
28  }
29  FUN_401097e8(0x3ffc4c3c,pcVar2);
30  return;
31 }
32
```

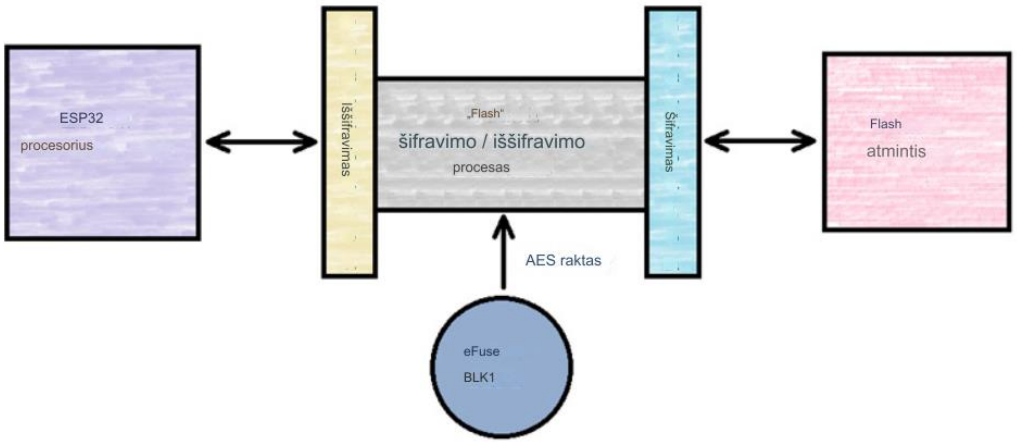
53 pav. Mikrovaldiklio prisijungimo taško dekompiliuota funkcija

```
void WiFiConfig()
{
  delay(5000);
  if (WiFi.status() == WL_CONNECTED && isSoftAPActive)
  {
    WiFi.mode(WIFI_STA);
    WiFi.softAPdisconnect(true);
    isSoftAPActive = false;
    Serial.println("Turned off SoftAP");
  }
  else if (WiFi.status() != WL_CONNECTED && !isSoftAPActive)
  {
    WiFi.mode(WIFI_AP_STA);
    WiFi.softAP(ap_ssid);
    isSoftAPActive = true;
    Serial.println("Started SoftAP");
  }
}
```

54 pav. Mikrovaldiklio prisijungimo taško funkcija

4.4. ESP32 "flash" atminties apsaugojimas

Norint sustiprinti „ESP32“ mikrovaldiklio atminties apsaugą, reikia atlikti „flash“ atminties šifravimą. „ESP32 Flash Encryption“ yra „Espressif System“ teikiama „ESP32“ apsaugos funkcija, skirta apsaugoti „flash“ atmintį [33]. Mikrovaldiklio iššifravimo atminties vykdymo procesas iššifruoja „flash“ atminties duomenis, kai „ESP32“ vykdymo blokas bando juos perskaityti, o rašymo proceso atveju šifravimo vykdymo procesas užšifruoja duomenis prieš juos įrašant į „flash“ atmintį.

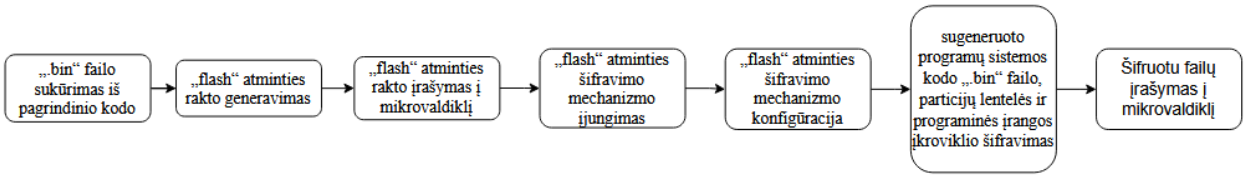


55 pav. „Flash“ atminties šifravimo/iššifravimo procesas

Šiuo procesu metu šifruojami šie skirsniai [33]:

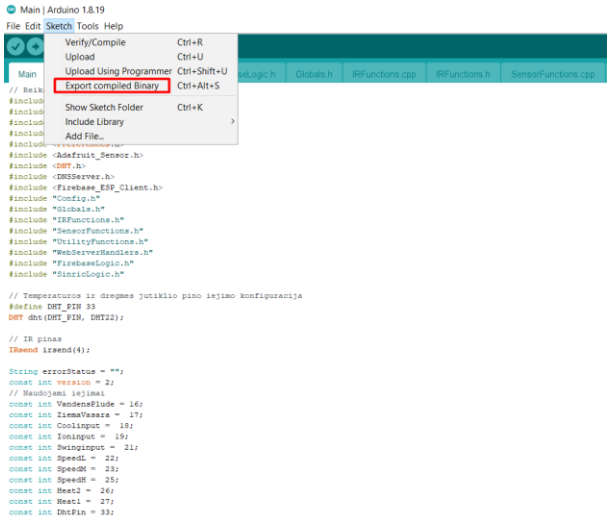
- programinės įrangos įkroviklis;
- skirsnių lentelė;
- „app“ tipo skirsniai arba tiesiog programų skirsniai.

Norint apsaugoti „ESP32“ mikrovaldiklio atmintį, visą procesą galime suskirstyti septynis žingsnius, kurie yra atvaizduoti srauto diagramoje.



56 pav. Šifravimo proceso srauto diagrama

Atminties šifravimui naudojami įrankiai „esptool.py“, „espefuse.py“ ir „espsecure.py“ [34]. Pirmiausia sukursime pagrindinio programų sistemų kodo „bin“ failą.



57 pav. Dvejetainio kodo rinkmenos sukūrimas

Komanda padedanti pasitikrinti ar „ESP32“ mikrovaldiklis jau turi sudėtus raktus. Komanda: „python -m espefuse --port "COM3" summary“. Jei matome nulius, tai reiškia, kad raktai nėra sudėti

```
BLOCK1 (BLOCK1)                                Flash encryption key
   = 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 R/W
BLOCK2 (BLOCK2)                                Security boot key
   = 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 R/W
```

58 pav. Šifravimo raktų vieta

Komanda generuojanti „flash“ atminties šifravimo raktus: „python -m espsecure generate_flash_encryption_key flash_raktas.bin“

```
C:\Users\Dell>python -m espsecure generate_flash_encryption_key flash_raktas.bin
espsecure.py v4.7.0
Writing 256 random bits to key file flash_raktas.bin
```

59 pav. Sugeneruotas šifravimo raktas

Komanda padedanti įrašyti „ESP32“ raktą: „python -m espefuse --port COM3 burn_key flash_encryption flash_raktas.bin“.

```
C:\Users\Dell>python -m espefuse --port COM3 burn_key flash_encryption flash_raktas.bin
espefuse.py v4.7.0
Connecting.........
Detecting chip type... Unsupported detection protocol, switching and trying again...
Connecting.....
Detecting chip type... ESP32

=== Run "burn_key" command ===
Sensitive data will be hidden (see --show-sensitive-info)
Burn keys to blocks:
 - BLOCK1 -> [?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ??]
        Reversing the byte order
        Disabling read to key block
        Disabling write to key block

Burn keys in efuse blocks.
The key block will be read and write protected


Check all blocks for burn...
idx, BLOCK_NAME,          Conclusion
[00] BLOCK0               is not empty
        (written ): 0x00000004011000000000002370000a200005948e7299ec64400000000
        (to write): 0x00000000000000000000000000000000000000000000000000010080
        (coding scheme = NONE)
[01] BLOCK1               is empty, will burn the new value
.
This is an irreversible operation!
Type 'BURN' (all capitals) to continue.
BURN
BURN BLOCK1  - OK (write block == read block)
BURN BLOCK0  - OK (all write block bits are set)
Reading updated efuses...
Successful
```

60 pav. Šifravimo rakto įrašymas

Komanda įjungianti „flash“ atminties šifravimo mechanizmą: „python -m espefuse --port COM3 burn_efuse FLASH_CRYPT_CNT 1“

```
C:\Users\Dell>python -m espefuse --port COM3 burn_efuse FLASH_CRYPT_CNT 1
espefuse.py v4.7.0
Connecting......
Detecting chip type... Unsupported detection protocol, switching and trying again...
Connecting....
Detecting chip type... ESP32

=== Run "burn_efuse" command ===
The efuses to burn:
  from BLOCK0
     - FLASH_CRYPT_CNT

Burning efuses:

    - 'FLASH_CRYPT_CNT' (Flash encryption is enabled if this field has an odd number of bits set) 0b0000000 -> 0b0000001


Check all blocks for burn...
idx, BLOCK_NAME,          Conclusion
[00] BLOCK0               is not empty
        (written ): 0x00000004011000000000002370000a200005948e7299ec64400010080
        (to write): 0x00000000000000000000000000000000000000000000000000100000
        (coding scheme = NONE)
.
This is an irreversible operation!
Type 'BURN' (all capitals) to continue.
BURN
BURN BLOCK0  - OK (all write block bits are set)
Reading updated efuses...
Checking efuses...
Successful
```

61 pav. Šifravimo mechanizmo įjungimas

Komanda konfigūruojanti „Flash“ atminties šifravimą taip, kad visi adreso bitai būtų naudojami kartu su šifravimo raktu (didžiausia vertė 0x0F)

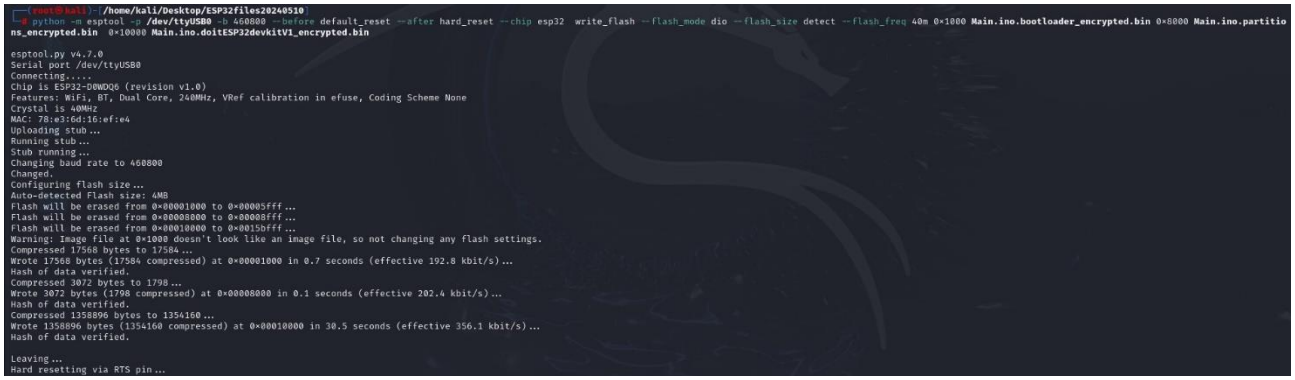
```
C:\Users\Dell>python -m espefuse --port COM3 burn_efuse FLASH_CRYPT_CONFIG 0x0F
espefuse.py v4.7.0
Connecting......
Detecting chip type... Unsupported detection protocol, switching and trying again...
Connecting....
Detecting chip type... ESP32

=== Run "burn_efuse" command ===
The efuses to burn:
  from BLOCK0
     - FLASH_CRYPT_CONFIG

Burning efuses:

    - 'FLASH_CRYPT_CONFIG' (Flash encryption config (key tweak bits)) 0x0 -> 0xf


Check all blocks for burn...
idx, BLOCK_NAME,          Conclusion
[00] BLOCK0               is not empty
        (written ): 0x00000004011000000000002370000a200005948e7299ec64400110080
        (to write): 0x0000000f0000000000000000000000000000000000000000000000000
        (coding scheme = NONE)
.
This is an irreversible operation!
Type 'BURN' (all capitals) to continue.
BURN
BURN BLOCK0  - OK (all write block bits are set)
Reading updated efuses...
Checking efuses...
Successful
```

62 pav. „Flash“ atminties šifravimo konfigūracija

Atliekame sugeneruoto programų sistemos kodo „.bin“ failo, particijų lentelės ir programinės įrangos įkroviklio šifravimą.

63 pav. failų šifravimas

Šifruotus failus įrašome į mikrovaldiklį panaudoja komandą: „python -m esptool -p /dev/ttyUSB0 -b 460800 --before default_reset --after hard_reset --chip esp32 write_flash --flash_mode dio --flash_size detect --flash_freq 40m 0x1000 Main.ino.bootloader_encrypted.bin 0x8000 Main.ino.partitions_encrypted.bin 0x10000 Main.ino.doitESP32devkitV1_encrypted.bin“



64 pav. failų įrašymas

Norint dar labiau papildyti apsaugą, galima panaudoti šias komandas:

- „python -m espefuse --port COM3 burn_efuse DISABLE_DL_ENCRYPT 0x1“;
- „python -m espefuse --port COM3 burn_efuse DISABLE_DL_DECRYPT 0x1“;
- „python -m espefuse --port COM3 burn_efuse DISABLE_DL_CACHE 0x1“.

Šios komandos prideda papildomas apsaugas mikrovaldikliui, kad „UART“ įkroviklis neiššifruotų "flash" atminties turinio.

Išgauname „flash“ atminties turinį panaudodami anksčiau naudotą komandą.

```
C:\Users\Dell>python -m esptool --baud 115200 --port COM3 read_flash 0x0 0x400000 FWsecure.bin
esptool.py v4.7.0
Serial port COM3
Connecting....
Detecting chip type... Unsupported detection protocol, switching and trying again...
Connecting....
Detecting chip type... ESP32
Chip is ESP32-D0WD (revision v1.0)
Features: WiFi, BT, Dual Core, 240MHz, VRef calibration in efuse, Coding Scheme None
Crystal is 40MHz
MAC: 80:7d:3a:08:8b:b5
Uploading stub...
Running stub...
Stub running...
4194304 (100 %)
4194304 (100 %)
Read 4194304 bytes at 0x00000000 in 379.6 seconds (88.4 kbit/s)...
Hard resetting via RTS pin...
```

65 pav. „flash“ atminties išgavimas

Įrankis „esp32knife“ nesukuria „.elf“ failo iš gauto „.bin“ failo, todėl analizei naudosime „.bin“ failą „Ghidra“ įrankyje.

```
┌──(kali㉿kali)-[~/Desktop/esp32knife-main]
└─$ python esp32knife.py --chip=esp32 load_from_file ./FWsecure.bin
Prepare output directories:
- creating directory: parsed
Reading firmware from: ./FWsecure.bin
<class 'esptool.FatalError'>
('Invalid firmware image magic=0x89',)
Invalid firmware image magic=0x89
```

66 pav. „esp32knife“ įrankio panaudojimas

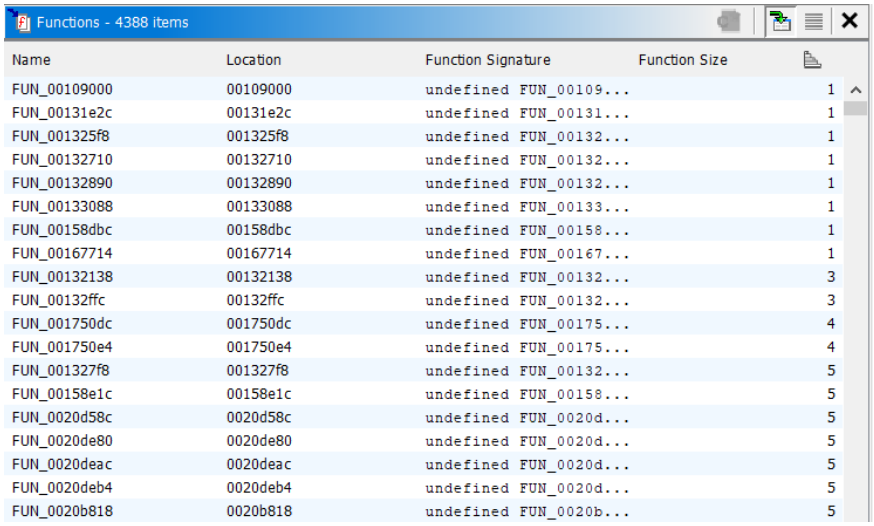
Atlikus analizuojamo sistemos failo analizę randame 46 simbolių eilučių.

Defined Strings - 46 items

Location	String Value	String Representation	Data Type
00009048	wifi_creds	"wifi_creds"	ds
000090a8	ap.sndchan	"ap.sndchan"	ds
000090c7	Popmode	"Popmode"	ds
00009108	cal_data	"cal_data"	ds
000098a8	cal_data	"cal_data"	ds
000098c5	K3Vcal_mac	"K3Vcal_mac"	ds
00009908	cal_mac	"cal_mac"	ds
00009927	qcal_version	"qcal_version"	ds
00009948	opmode	"opmode"	ds
00009967	yap.authmode	"yap.authmode"	ds
00009986	m2ap.chan	"m2ap.chan"	ds
000099a8	ap.ssid	"ap.ssid"	ds
000099c4	SmartHeatAP	"SmartHeatAP"	ds
00009a07	kap.ssid	"kap.ssid"	ds
00009a28	ap.passwd	"ap.passwd"	ds
00009aa8	ap.passwd	"ap.passwd"	ds
00009ac7	wap.hidden	"wap.hidden"	ds
00009ae7	Yap.max.conn	"Yap.max.conn"	ds
00009b06	]Vbcn.interval	"]Vbcn.interval"	ds

67 pav. Rastos simbolių eilutės

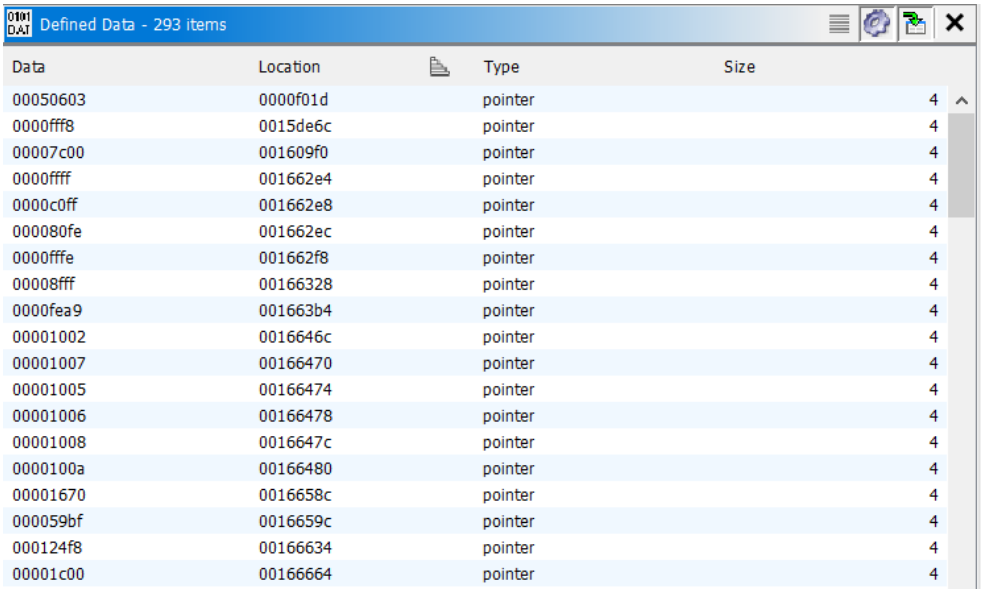
Atlikus analizuojamo sistemos failo analizę randame 4388 funkcijas.


Functions - 4388 items

Name	Location	Function Signature	Function Size
FUN_00109000	00109000	undefined FUN_00109...	1
FUN_00131e2c	00131e2c	undefined FUN_00131...	1
FUN_001325f8	001325f8	undefined FUN_00132...	1
FUN_00132710	00132710	undefined FUN_00132...	1
FUN_00132890	00132890	undefined FUN_00132...	1
FUN_00133088	00133088	undefined FUN_00133...	1
FUN_00158dbc	00158dbc	undefined FUN_00158...	1
FUN_00167714	00167714	undefined FUN_00167...	1
FUN_00132138	00132138	undefined FUN_00132...	3
FUN_00132ffc	00132ffc	undefined FUN_00132...	3
FUN_001750dc	001750dc	undefined FUN_00175...	4
FUN_001750e4	001750e4	undefined FUN_00175...	4
FUN_001327f8	001327f8	undefined FUN_00132...	5
FUN_00158e1c	00158e1c	undefined FUN_00158...	5
FUN_0020d58c	0020d58c	undefined FUN_0020d...	5
FUN_0020de80	0020de80	undefined FUN_0020d...	5
FUN_0020deac	0020deac	undefined FUN_0020d...	5
FUN_0020deb4	0020deb4	undefined FUN_0020d...	5
FUN_0020b818	0020b818	undefined FUN_0020b...	5

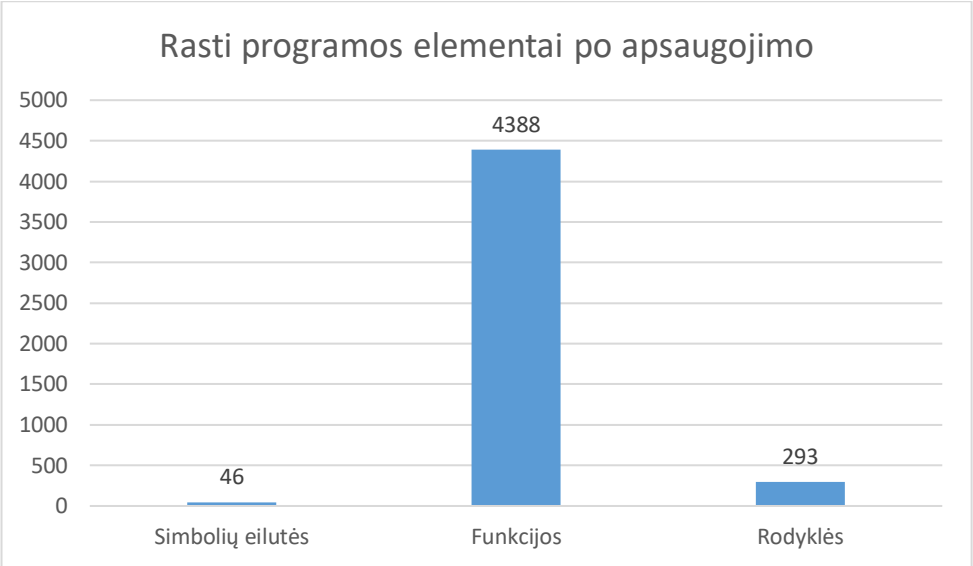
68 pav. Rastos funkcijos

Atlikus analizuojamo sistemos failo analizę randame 293 rodykles.


Defined Data - 293 items

Data	Location	Type	Size
00050603	0000f01d	pointer	4
0000fff8	0015de6c	pointer	4
00007c00	001609f0	pointer	4
0000ffff	001662e4	pointer	4
0000c0ff	001662e8	pointer	4
000080fe	001662ec	pointer	4
0000fffe	001662f8	pointer	4
00008fff	00166328	pointer	4
0000fea9	001663b4	pointer	4
00001002	0016646c	pointer	4
00001007	00166470	pointer	4
00001005	00166474	pointer	4
00001006	00166478	pointer	4
00001008	0016647c	pointer	4
0000100a	00166480	pointer	4
00001670	0016658c	pointer	4
000059bf	0016659c	pointer	4
000124f8	00166634	pointer	4
00001c00	00166664	pointer	4

69 pav. Rastos rodyklės



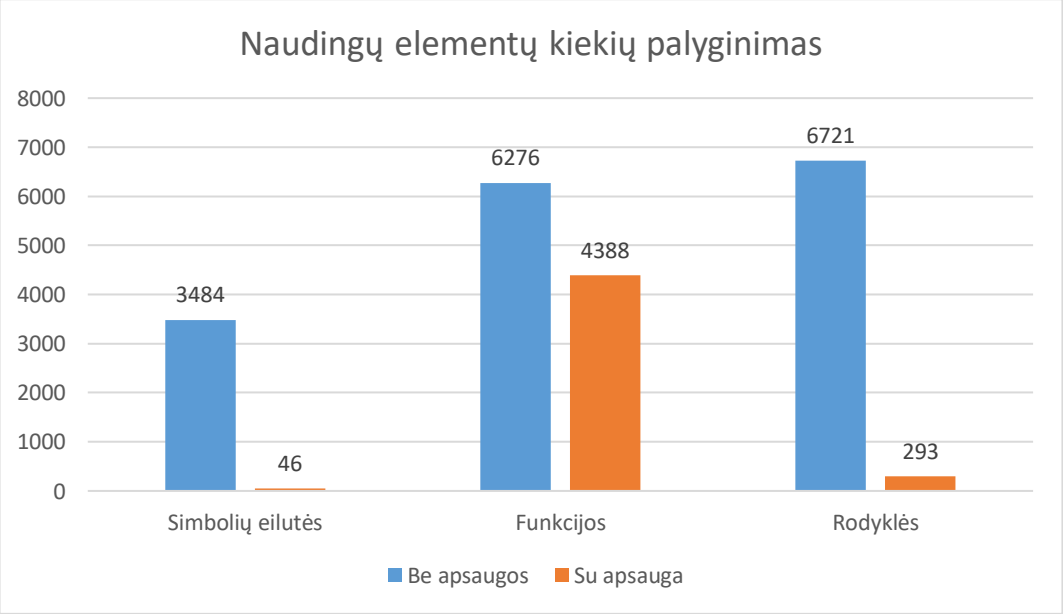
70 pav. Rasti programos elementai panaudojant „Ghidra“

4.5. Gautu rezultatų palyginimas

Po „flash“ šifravimo randamų naudingų elementų kiekis pasikeitė:

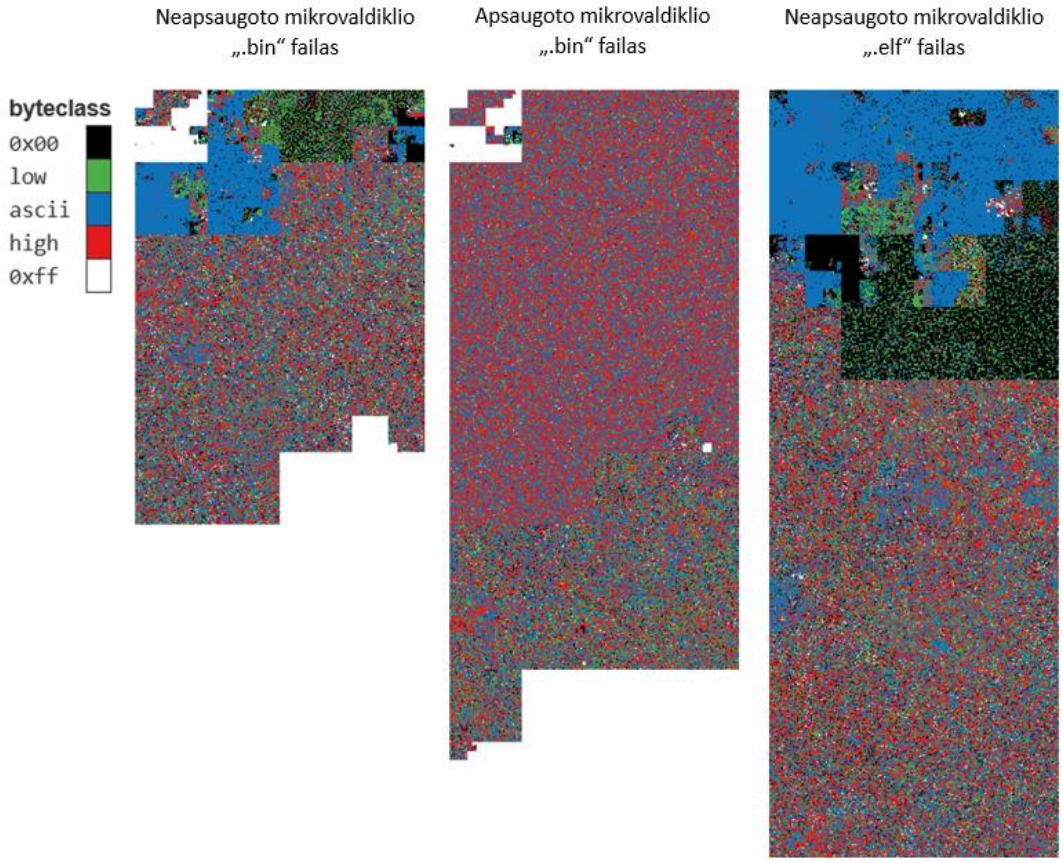
- randamu simbolių eilučių kiekis sumažėjo 3438 vienetais arba 98,68 %;
- randamu funkcijų kiekis sumažėjo 1888 vienetais arba 30,07 %;
- randamu rodyklių kiekis sumažėjo 6428 vienetais arba 95,64 %.

Žiūrint į šiuos rezultatus galime teigti, kad randamos naudingos informacijos kiekis sumažėjo atlikus „flash“ atminties šifravimą.



71 pav. Programų sistemos elementų kiekių palyginimas

Norint vizualizuoti gautus „.bin“ ir „.elf“ failus galime naudoti „binvis.io“ tinklalapį. Atlikus vizualiciją matome, kad kairysis stulpelis, kuris yra neapsaugoto mikrovaldiklio „.bin“ failas, rodo gana tolygų įvairių duomenų tipų pasiskirstymą, o tai rodo, kad šifravimo nėra. Vidurinysis stulpelis, kuris yra apsaugoto mikrovaldiklio „.bin“ failas, rodo, kad duomenų tipų pasiskirstymas pasikeitė - daugiau „didelės“ entropijos sričių rodo šifravimą, o dešinysis stulpelis, kuris yra neapsaugoto mikrovaldiklio „.elf“ failas, rodo aiškesnę struktūrą, kuri yra būdinga „.elf“ failams ir tai rodo, kad šifravimo nėra.



72 pav. „bin“ ir „.elf“ failų vizualizacija

4.6. Tyrimo išvados

Atliktas tyrimas išbandant „ESP32“ mikrovaldiklio „flash“ atminties šifravimą. Peržiūrėję gautus rezultatus matome:

1. randamų simbolių eilučių kiekis sumažėjo 98,68 %;
2. randamų funkcijų kiekis sumažėjo 30,07 %;
3. randamų rodyklių kiekis sumažėjo 95,64 %.

Tai rodo, kad atminties šifravimas padeda apsaugoti reikšmingą programų sistemos informaciją nuo galimo sistemos puolėjo arba atvirkštinės inžinerijos inžinieriaus. Norint dar labiau apsaugoti informaciją, galima naudoti naujesnę mikrovaldiklio „EPS32“ konfigūraciją – „ESP32 - S3“, kuri turi daugiau papildomų apsaugos funkcijų.

Išvados

1. Literatūros analizės metu išnagrinėta kaip atvirkštinė inžinerija yra taikoma programinės įrangos inžinerijoje, elektronikos inžinerijoje, mechanikos inžinerijoje, chemijos inžinerijoje. Išnagrinėta kokie yra atvirkštinės inžinerijos įrankiai yra naudojami programinės įrangos inžinerijoje. Išnagrinėti būdai kaip apsisaugoti nuo atvirkštinės inžinerijos skirtingose inžinerijos srityse.
2. Sukurta programų sistema, kuria sudaro oro vėsintuvas-šildytuvas, „ESP32“ mikrovaldiklis, serveris, papildomi jutikliai, relės ir infraraudonųjų spindulių siųstuvas.
3. Tyrimo metu atlikta mikrovaldiklio programos kodo statinė analizė ir atlikti patobulinimai, kurie padeda sumažinti programos dydį 48 baitais, taip sutaupant vietos mikrovaldiklyje.
4. Eksperimento metu atlikti bandymai su neapsaugotu mikrovaldikliu ir apsaugotu mikrovaldikliu atliekant atminties šifravimą. Gauti rezultatai rodo, kad randamų simbolių eilučių kiekis sumažėjo 98,68 %, randamų funkcijų kiekis sumažėjo 30,07 % ir randamų rodyklių kiekis sumažėjo 95,64 %. Tai leidžia suprasti, kad „flash“ atminties šifravimas padeda apsaugoti naudingus programų sistemos elementus.

Literatūros sąrašas

1. Sachin Bhatnagaršaltinis, „What Is Reverse Engineering in Software Engineering?“. Prieiga internete: https://www.knowledgehut.com/blog/web-development/reverse-engineering-in-software-engineering [žiūrėta 2024 04 02]
2. Muhammad Muzammul, Muhammad Awais, „An empirical approach for software reengineering process with relation to quality assurance mechanism“. Prieiga internete: https://www.researchgate.net/figure/Reverse-and-forward-engineering-46_fig1_329803475 [žiūrėta 2024 04 02]
3. Eugenio de Tomaso, „Reverse Engineering: What It Is and How It Works“. Prieiga internete: https://www.codepwr.com/reverse-engineering-what-it-is-and-how-it-works/ [žiūrėta 2024 04 02]
4. Chiradeep BasuMallick, „What Is Reverse Engineering? Definition, Working, Examples, and Importance“. Prieiga internete: https://www.spiceworks.com/tech/tech-general/articles/what-is-reverse-engineering/ [žiūrėta 2024 04 02]
5. Ralph Meier, „Reverse Engineering Introduction to the World of Disassembling and Decompiling“. Prieiga internete: https://www.scip.ch/en/?labs.20211202 [žiūrėta 2024 04 02]
6. Priyadharshini Balaji, „The Interactive Disassembler – IDAPro“. Prieiga internete: https://www.socinvestigation.com/the-interactive-disassembler-ida-pro/ [žiūrėta 2024 04 02]
7. Neil Fox, „How to Use Ghidra to Reverse Engineer Malware“. Prieiga internete: https://www.varonis.com/blog/how-to-use-ghidra [žiūrėta 2024 04 02]
8. Emil Hozan, „Android APK Reverse Engineering: Using JADX“. Prieiga internete: https://www.secplicity.org/2019/10/04/android-apk-reverse-engineering-using-jadx/ [žiūrėta 2024 04 02]
9. Peter Vogel, „Review: Peek at Someone Else's Code with Red Gate Reflector“. Prieiga internete: https://visualstudiomagazine.com/articles/2011/07/01/pdvst_reflector.aspx [žiūrėta 2024 04 02]
10. Roger Hart, Carlos Quintero, „Guest post: Using .NET Reflector to understand and debug Visual Studio assemblies“ .Prieiga internete: https://www.red-gate.com/products/reflector/guest-post [žiūrėta 2024 04 09]
11. William Largent, „Vulnerability Spotlight: Hopper Disassembler ELF Section Header Size Code Execution“. Prieiga internete: https://blog.talosintelligence.com/hopper/ [žiūrėta 2024 04 09]
12. Tapas Pal, „DotPeek Tool: A .NET Decompiler“ Prieiga internete: https://www.codeguru.com/dotnet/dotpeek-tool-a-net-decompiler/ [žiūrėta 2024 04 09]
13. Dr. Narayan Joshi, „GNU Debugger (GDB)“. Prieiga internete: https://ebooks.inflibnet.ac.in/itp7/chapter/gnu-debugger-gdb/ [žiūrėta 2024 04 09]
14. Aditya Trivedi, „PDB Python Debugger“. Prieiga internete: https://www.scaler.com/topics/pdb-python/ [žiūrėta 2024 04 09]
15. „MarketSplash“ komanda, „How To Utilize Rubymine Debugger For Efficient Coding“. Prieiga internete: https://marketsplash.com/tutorials/all/rubymine-debugger/ [žiūrėta 2024 04 09]
16. Avi Tsadok, „Advanced Debugging — Part 1: LLDB Console“. Prieiga internete: https://www.linkedin.com/pulse/advanced-debugging-part-1-lldb-console-avi-tsadok/ [žiūrėta 2024 04 09]
17. Md Tarikul Islam Sheam, „Reverse Engineering in a simple eye“. Prieiga internete: https://www.linkedin.com/pulse/reverse-engineering-simple-eye-md-tarikul-islam-sheam/ [žiūrėta 2024 04 15]

18. RayMingPCB, „The Complete Guide to Electronic Reverse Engineering“. Prieiga internete: https://medium.com/@raymingpcb/the-complete-guide-to-electronic-reverse-engineering-69a6446281a5 [žiūrėta 2024 04 15]
19. Julia Zhang, „Circuit Board Tester And Method Conducive To PCB Repair“. Prieiga internete: https://www.linkedin.com/pulse/circuit-board-tester-method-conducive-pcb-repair-julia-z--bkobc/ [žiūrėta 2024 04 15]
20. Raja Sekhar Upputuri, „What is Reverse Engineering and its Applications“ . Prieiga internete: https://www.think3d.in/what-is-reverse-engineering-and-its-applications/ [žiūrėta 2024 04 15]
21. Michal Dúbravčík, Štefan Kender, „Application of Reverse Engineering Techniques in Mechanics System Services“. Prieiga internete: https://www.sciencedirect.com/science/article/pii/S1877705812045547 [žiūrėta 2024 04 15]
22. Redakcijos komanda, „Introduction to chemical reverse engineering; methods and use“. Prieiga internete: https://onlytrainings.com/introduction-to-chemical-reverse-engineering-methods-and-use-onlytrainings-blog [žiūrėta 2024 04 15]
23. Kamei Nao, „Protecting Your Software from Reverse Engineering: Strategies and Best Practices“ “. Prieiga internete: https://kameinao.medium.com/protecting-your-software-from-reverse-engineering-strategies-and-best-practices-2533d83d9749 [žiūrėta 2024 04 15]
24. Bowen Zhang, „Research Summary of Anti-debugging Technology“ . Prieiga internete: https://iopscience.iop.org/article/10.1088/1742-6596/1744/4/042186/pdf [žiūrėta 2024 04 15]
25. Nthatisi Hlapisi, „Protect PCBA Designs From Reverse Engineering: Tips and Tricks“ . Prieiga internete: https://www.macrofab.com/blog/protect-pcba-designs-reverse-engineering/ [Žiūrėta 2024 04 17]
26. Will Li, „PCB Potting vs. Conformal Coating – Which One Is Better?“ . Prieiga internete: https://www.mokotechnology.com/pcb-potting-vs-conformal-coating/ [Žiūrėta 2024 04 17]
27. Ross, J.P. & Meier, Alan, „Whole-house measurements of standby power consumption“. Prieiga internete: https://digital.library.unt.edu/ark:/67531/metadc743210/ [Žiūrėta 2024 04 17]
28. Florian Mueller, „Achieving very low or zero standby power for AC-DC power supplies“ . Prieiga internete: https://eepower.com/technical-articles/achieving-very-low-or-zero-standby-power-for-acdc-power-supplies/ [Žiūrėta 2024 04 17]
29. Renzo Mischianti, „DOIT ESP32 DEV KIT v1: high resolution pinout and specs“ https://mischianti.org/doit-esp32-dev-kit-v1-high-resolution-pinout-and-specs/ [Žiūrėta 2024 04 17]
30. Daniel Marjamäki, „Cppcheck - A tool for static C/C++ code analysis Overview“. Prieiga internete: https://sourceforge.net/p/cppcheck/wiki/Home/ [Žiūrėta 2024 04 17]
31. Achim Pieters, „ESP32 – ESPTool“. Prieiga internete: https://www.studiopieters.nl/esp32-esptool/ [Žiūrėta 2024 04 17]
32. Pedro Minatel, „How to use custom partition tables on ESP32“ . Prieiga internete: https://blog.espressif.com/how-to-use-custom-partition-tables-on-esp32-69c0f3fa89c8 [Žiūrėta 2024 04 17]
33. Noyel Seth, „How to Secure ESP32 Firmware and Flash Memory on ESP-IDF Framework“. Prieiga internete: https://circuitdigest.com/article/how-to-secure-esp32-firmware-and-flash-memory-using-esp-idf-framework [Žiūrėta 2024 04 17]
34. Roshan Kattel, „ESP32 Firmware Encryption“ “. Prieiga internete: https://medium.com/@kattelroshan1/esp32-firmware-encryption-a53eb1c9bf9c [Žiūrėta 2024 04 17]