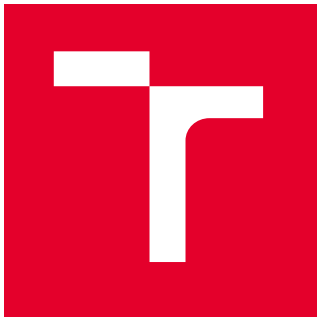


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV TELEKOMUNIKACÍ

DEPARTMENT OF TELECOMMUNICATIONS

ZABEZPEČENÍ EDGE SENZORŮ

SECURITY OF EDGE SENSORS

BAKALÁŘSKÁ PRÁCE

BACHELOR'S THESIS

AUTOR PRÁCE

AUTHOR

David Jančík

VEDOUCÍ PRÁCE

SUPERVISOR

Ing. Petr Ilgner

BRNO 2022

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Bakalářská práce

bakalářský studijní program **Informační bezpečnost**

Ústav telekomunikací

Student: David Jančík ***ID:*** 223326

Ročník: 3 ***Akademický rok:*** 2021/22

NÁZEV TÉMATU:

Zabezpečení EDGE senzorů

POKYNY PRO VYPRACOVÁNÍ:

Seznamte se s možnými způsoby zabezpečení senzorů založených na platformě Raspberry Pi. Navrhněte a realizujte návrh zabezpečení těchto senzorů, a to s možností využití SAM modulů.

Výstupem práce bude návrh a realizace řešení pro zabezpečení EDGE senzorů zajišťující: 1) zabezpečení citlivých dat na disku pomocí šifrování a SAM modulu, 2) zabezpečení senzoru proti neoprávněnému fyzickému vniknutí a 3) možnost řízení přístupu do senzoru pomocí Android mobilní aplikace.

DOPORUČENÁ LITERATURA:

[1] MENEZES, Alfred, Paul C VAN OORSCHOT a Scott A VANSTONE. Handbook of applied cryptography. Boca Raton: CRC Press, c1997. Discrete mathematics and its applications. ISBN 0-8493-8523-7.

[2] K. Sha, R. Errabelly, W. Wei, T. A. Yang and Z. Wang, "EdgeSec: Design of an Edge Layer Security Service to Enhance IoT Security," 2017 IEEE 1st International Conference on Fog and Edge Computing (ICFEC), 2017, pp. 81-88, doi: 10.1109/ICFEC.2017.7.

Termín zadání: 7.2.2022 ***Termín odevzdání:*** 31.5.2022

Vedoucí práce: Ing. Petr Ilgner

doc. Ing. Jan Hajný, Ph.D.
předseda rady studijního programu

ABSTRAKT

Tato bakalářská práce se zabývá zabezpečení EDGE senzorů, na platformě Raspberry Pi, který slouží k uchování citlivých dat na disku. V této práci je testovací zařízení Raspberry Pi 4. Nejprve se zkoumá a implementuje na zařízení zabezpečovací modul SD karta od společnosti Swissbit, Swissbit PS-45u, a Zymkey 4i od společnosti Zymbit. Na základě získaných poznatků Zymkey 4i a Swissbit dochází k návrhu a implementaci vlastního fyzického bezpečnostního modulu. Autentizace oprávněného uživatele je vyřešena pomocí čipových karet, které podporují NFC komunikaci. Vlastní fyzický bezpečnostní modul využívá I^2C komunikaci, která je implementována na prototypové desce. Komponenty napájené na prototypové desce jsou Arduino Nano, akcelerometr a gyroskop, dva propojovací dráty, dvě světelné diody a NFC rozhraní.

KLÍČOVÁ SLOVA

Swissbit, Zymbit, zabezpečení EDGE senzorů, Raspberry Pi 4, I^2C komunikace, Arduino Nano, prototypová deska, IoT, NFC

ABSTRACT

This bachelor thesis deals with the security of EDGE sensors on the Raspberry Pi platform, which is used to store sensitive data on disk. In this thesis, the test device is Raspberry Pi 4. First, it examines and implements an SD card security module from Swissbit, Swissbit PS-45u, and Zymkey 4i from Zymbit. With the knowledge of Zymkey i4 and Swissbit to design and implement its own physical security module. Authentication is handled using smart cards that support the NFC communication. The components powered on the prototype board are an Arduino Nano, an accelerometer and a gyroscope, two connecting wires, two LEDs and an NFC interface.

KEYWORDS

Swissbit, Zymbit, EDGE sensor security, Raspberry Pi 4, I^2C communication, Arduino Nano, prototype board, IoT, NFC

Vysázeno pomocí balíčku `thesis` verze 4.07; `http://latex.feec.vutbr.cz`

Prohlášení autora o původnosti díla

Jméno a příjmení autora:	David Jančík
VUT ID autora:	223326
Typ práce:	Bakalářská práce
Akademický rok:	2021/22
Téma závěrečné práce:	Zabezpečení EDGE senzorů

Prohlašuji, že svou závěrečnou práci jsem vypracoval samostatně pod vedením vedoucí/ho závěrečné práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené závěrečné práce dále prohlašuji, že v souvislosti s vytvořením této závěrečné práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a/nebo majetkových a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

Brno ....

podpis autora*

*Autor podepisuje pouze v tištěné verzi.

PODĚKOVÁNÍ

Tímto bych chtěl poděkovat panu Ing. Petrovi Ilgnerovi, za odborné vedení bakalářské práce a za ochotu vždy pomoci a poradit při jejím vypracování.

Obsah

Seznam obrázků

Seznam výpisů

Úvod

V dnešní době je IoT, zkratka pro Internet of Things, v českém překladu Internet věcí, běžnou věcí. V současné době je využití prvků Internetu věcí (IoT) běžné – od klimatizace, kterou lze ovládat našimi chytrými telefony nebo při kontrole stavu paliva či zajišťování diagnostiky poruch v automobilech až po užívání chytrých hodinek, které sledují denní aktivity a zdravotní funkce. Každé zařízení má vlastní senzory, jež pomohou detekovat a reagovat na změny prostředí, dále se zde řadí chytré domácí spotřebiče nebo také snímače čárových kódů, dopravních semaforů a mnoho jiných, dalších zařízení. Předpokládá se, že v roce 2025 má být až 30,9 miliard funkčních zařízení na světe [1].

Tato práce se zabývá zabezpečováním senzorů, jelikož IoT je obsažen v nespočetně fyzických zařízení, u kterých se předpokládá každoroční nárůst a nebere se na zabezpečovací senzory takový důraz, jakého by bylo potřeba. IoT je zapotřebí chránit před neoprávněnými uživateli nebo útočníky, aby data, která jsou uložená v zařízení, nebyla odcizena.

Úvod práce bude pojednávat o ochraně těchto zařízení. Nejstěžnějším zařízením, na kterém budou pokusy probíhat, bude zařízení Raspberry Pi 4 model B, což je jednodeskový počítač, který bude shromažďovat všechna data, která se později uchovají a začnou chránit.

Nejprve bude použit hardwarový zabezpečovací modul Zymkey 4i od společnosti Zymbit. Má nespočet přínosných vlastností, jako je např. více faktorová identita zařízení a autentizace, šifrovaní či podepisování, fyzické senzory pro detekci neoprávněné manipulace Používá sadu silných šifer jako jsou ECDSA, ECDH, AES-256, SHA-256 [2].

Poté bude následovat využití microSD Card Swissbit PS-45u, která dokáže automaticky šifrovat a chránit přístup pro veškerý uložený obsah, jako jsou fotografie, videa a data. MicroSD Card Swissbit PS-45u využívá šifru AES-256. [3].

Hlavním cílem této práce je navrhnout zabezpečení senzorů založených na platformě Raspberry Pi, které bude umožňovat zabezpečit citlivá data na disku pomocí šifrování a SAM modulu. Dalším krokem bude zabezpečení senzorů proti neoprávněnému fyzickému vstupu. A v neposlední řadě možnost řízení přístupu pomocí čipové kartě.

1 Bezpečnost IoT zařízení

IoT je síť fyzických zařízení, která jsou jednotlivě propojená v dané síti a všechny tyto zařízení dokáží komunikovat, shromažďovat, ukládat a sdílet určitá data. Internet věcí je velmi rozmanitý pojem, který usnadňuje spoustu věcí a lze jej využít v odlišných odvětvích a dosáhnout tak k vytvoření chytrých domácností, modernějších měst, zemědělství, zdravotnictví, armády a mnoho dalších odvětví [4].

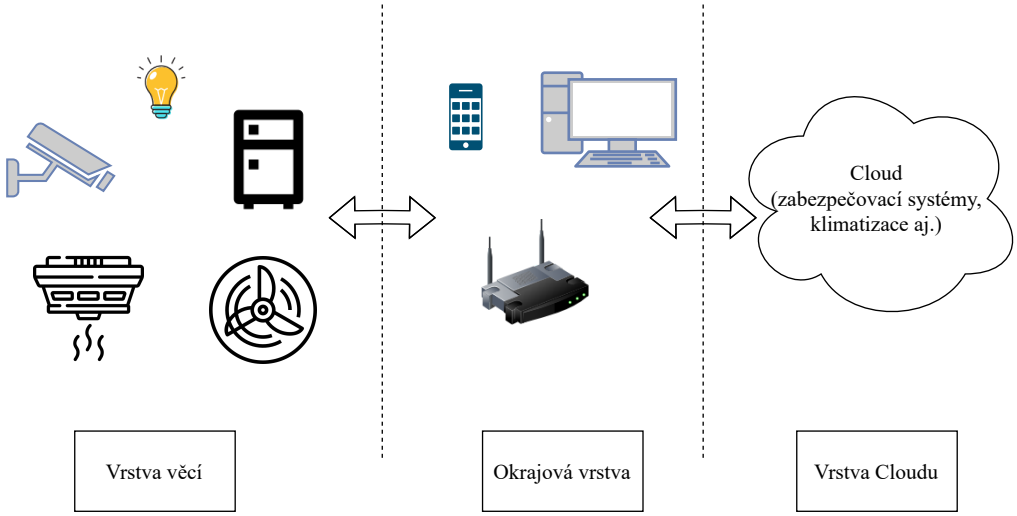
Je velmi důležité dbát na možná bezpečnostní rizika, která mohou ve spojení s IoT nastat, protože použitá zařízení mají často bezpečnostní zranitelnosti, které nejsou objeveny nebo výrobcem zařízení v rámci životního cyklu produktu opraveny, případně nejsou tyto záplaty aplikovány uživateli těchto zařízení. V následujících podkapitolách je rozebráno, jaké nebezpečí hrozí a jakým způsobem Internet věcí usnadňuje každodenní život.

1.1 Chytrá domácnost

Internet věcí nám v domácnostech přináší spoustu benefitů. Díky těmto benefitům je možné mít domov pod dohledem, protože poskytují možnosti správy nastavení připojených zařízení, což ušetří nespočet času i práce. Je tedy možné ovládat a řídit televizi, počítač, ledničku, zabezpečovací zařízení, osvětlení, vysavač, termostat a další elektroniku nezbytnou v každé domácnosti. Pomocí IoT zařízení lze chytrou domácnost také automatizovat, například tak, že rozpozná přiblížení auta majitele domu a pomocí sensorů se automaticky otevře garáž či se rozsvítí dům při příchodu tmy [5].

Na obr. 1.1 je třívrstvá architektura chytré domácnosti dle [5]. Levá část vyobrazuje možné zaznamenání vrstvy Things Layer (tzv. Vrstva věcí). Zde patří např. kamery ale i detektor kouře. Prostřední segment představuje Edge Layer (tzv. Okrajovou vrstvu), kam spadají routery, telefony a další zařízení, která pomáhají připojit koncová zařízení k internetu. Pravá část je Cloud Layer (tzv. Vrstva Cloudu), kde se shromažďují a ukládají data. Další funkcí Cloud Layeru je podpora pro další aplikace, tj. klimatizace, osvětlení, zabezpečovací zařízení [5].

Současný trend IoT vede k faktu, že výrobci se snaží vytvořit co nejvíce produktů podporující Internet věcí. Některá zařízení neprošla dostatečnými bezpečnostními testy, což může mít za následek, že tato zařízení jsou potenciálně zranitelná. Nové produkty nahrazují starší, bohužel však starší zařízení postrádají aktualizace od samotných vývojářů. Důsledkem přehlížení starších zařízení ze strany vývojářů ke starším zařízením se stává, že pokud je objevená nějaká zranitelnost, tak tam bude stále setrvávat, jelikož nejsou vytvořeny žádné nové aktualizace či příliš dlouho trvá, než je vydána nová aktualizace. Tím je i pro méně zkušené hackery snazší proniknout



Obr. 1.1: Třívrstvá architektura chytré domácnosti [5].

do sítí domácností, protože jejich součástí může být i starší elektronika, typu televize nebo lednice, které jsou dlouhodobě neaktualizované [5, 6, 7].

Hacker může kvůli zastaralým zařízením nebo nedostatečné ochraně těchto zařízení, získat přístup do systémového vytápění a osvětlení, díky tomu si opatřit hesla od jednotlivých účtů, a to prostřednictvím informací, které jsou sdíleny v počítačových nebo jiných zařízení v síti. Prostřednictvím zastaralého zařízení, což nejčastěji představují spotřebiče, se dostane do sítě a můžete ovládat celý dům nebo spustit tzv. Ransomwarový útok, díky kterému se chytrá domácnost stává nepoužitelnou, a to do doby splnění podmínek útočníka [5, 6, 7].

1.2 Chytré město

Internet věcí se také hojně využívá v chytrých městech, pomocí něhož lze dosáhnout lepší plynulosti dopravy, zlepšení energetickou účinnost a zdraví obyvatel, modernizování infrastruktury, zlepšení spolupráce občanů či zvýšení počtu bezpečnostních kamer snímajících jak občany, tak mezinárodní poznávací značky. Tyto kroky společně ruku v ruce vedou ke zvýšené bezpečnosti města [8].

Jedny z největších zranitelností chytrých měst tkví v technice tzv. Man-in-the-middle, tedy odposlouchávání komunikaci mezi dvěma zařízeními nebo tzv. hijacking, kdy útočník přebere kontrolu nad zařízením. Krom toho však existuje i riziko získání fyzického přístupu útočníka k zařízení. Existují zařízení, která jsou nasazována v bezobslužných prostředích, tím umožňují neoprávněné osobě extrahovat informace, jako jsou např. šifrovací klíče, uživatelská jména a hesla, osobní

údaje, obrázky [9].

1.3 Chytrá nemocnice

Pomocí IoT mohou nemocnice využívat nových prostředků jak pro zdravotníky, tak pro pacienty. Jednou z nejběžnějších aplikací je sledování a monitorování pacientů ze strany doktorů, ale i sledování pacientova vlastního zdraví jim samotným. Díky tomu lze pozorovat změny v těle – zvýšení teploty, změnu krevního tlaku či srdeční frekvence a jiných veličin [10].

Jeden z velkých problémů chytrých nemocnic je fakt, že zařízení pro monitorování pacientů nejsou aktuální. Nemocnice většinou používají starší verze softwaru nebo firmwaru nebo také využívají starých zařízeních, která jsou používána i po skončení životního cyklu zařízení a tedy skočení jejich podpory výrobcem. Tyto nedostatky pak mohou usnadnit případný útok do sítě a následné zcizení chráněných informací o pacientech [10].

1.4 Komunikační protokoly v IoT

Komunikační protokoly poskytují důležitou funkci v Internetu věcí, jelikož zodpovídají za navazování spojení mezi různými koncovými zařízeními. Mohou kontrolovat autorizace a detekovat chyby v komunikačních kanálech, poskytovat přístup k internetu, přenášet soubory a odesílat zprávy.

Jelikož veškerá komunikace probíhá přes komunikační protokoly, je zásadní používat specifické komunikační protokoly, které umožňují zajištění bezpečnosti a integrity přenášených dat, autentizace uživatelů a zařízení a tedy neumožňují neoprávněným uživatelům přistupovat k přenášeným informací.

Níže jsou uvedeny příklady, zabezpečených a jaké nezabezpečených protokolů.

SSH

SSH (Secure Shell) je zabezpečený komunikační protokol přes nezabezpečenou síť v architektuře klient-server, který nahrazuje Telnet, FTP a další nezabezpečené protokoly. Zabraňuje jakémukoliv neoprávněnému uživateli, aplikaci, službě nebo zařízení v přístupu k síťovým datům, jelikož komunikace je šifrovaná. Používá veřejný klíč k tomu, aby ověřil, zda se jedná o skutečné vzdálené zařízení. Pokud se uživatel ověří soukromým klíčem, zahájí se komunikace mezi dvěma rozhraními.

Wi-Fi

Wi-Fi je bezdrátový zabezpečovací protokol založený na standartu IEEE 802.11. Je to jeden z nejpopulárnějších komunikačních protokolů IoT. Poskytuje připojení k internetu blízkým zařízením v určitém rozsahu a nabízí rychlý přenos dat. Telefony nebo počítače mohou sdílet bezdrátové připojení k internetu. Wi-Fi vysílá rádiové vlny v konkrétních frekvencích jako jsou kanály 2,4 GHz nebo 5 GHz. IEEE 802.11 poskytuje zabezpečení pomocí šifrování a ověřování. Je velmi vhodný pro přenos velkých množství dat.

Efektivnímu zabezpečení IoT zařízení lze dosáhnout používáním silných hesel a vhodných autentizačních a šifrovacích algoritmů na přístupovém bodu. Příkladem opatření pro bezpečnou bezdrátovou síť je použití silné metody šifrování jako je WPA2 či WPA3.

Bluetooth

Bluetooth je jednou z nejdůležitějších komunikačních technologií na krátkou vzdálenost. Je vhodný pro menší množství dat, jelikož jeho rychlost přenosu je daleko nižší než Wi-Fi. Jde o otevřený standard bezdrátové technologie pro přenos dat, který dokáže propojovat dvě a více zařízení. Bluetooth je zabezpečený komunikační protokol, který podporuje ověřování a šifrování. Tyto funkce jsou založeny na soukromém odkazu klíčů, který sdílí dvojice zařízení. Jestliže nebyly ještě předtím klíče vygenerovány nastane operace párování zařízení.

V dnešní době se hodně využívá Bluetooth Low-Energy nebo také Bluetooth v IoT. Je to z důvodu poskytování snížené spotřební energii, dokáže vydržet měsíce či roky, mají nízké náklady, a pracují na podobné bázi jako Bluetooth.

Telnet

Telnet (Teletype Network) je starší komunikační protokol sloužící uživatelům k připojení ke svému vzdálenému zařízení pomocí textového uživatelského rozhraní. Nepoužívá šifrování, tudíž veškerá komunikace je v prostém textu, stejně jako přihlašovací údaje. To znamená, že kdokoliv, kdo má v síti spuštěný analyzátor paketu, bude následně schopen zachytit informace potřebné k přebrání kontroly nad daným zařízením, a to během několika sekund.

FTP

FTP (File Transfer Protocol) slouží k přenosu souborů mezi dvěma rozhraními. Patří k nebezpečným protokolům, protože je stavěn na ověřování uživatelského jména a

hesla v prostém textu. Pomocí paket analyzátoru lze zpětně zjistit autentizace, tedy jméno a heslo uživatele, protože tyto údaje nejsou nijak chráněny.

HTTP

HTTP (Hypertext Transfer Protocol) funguje na bázi klient-server jako předešlé protokoly, které nejsou řádně zabezpečeny. Nepoužívá šifrování, a tím pádem může jakýkoliv útočník získat pomocí analyzátoru paketu uživatelské jméno a heslo.

1.5 Zranitelnosti a prevence bezpečnosti IoT

Zranitelností Internetu věcí je nejčastěji využití slabých hesel nebo hesel nastavených výrobcem, nedostatečně zabezpečené síťové služby, nezabezpečené rozhraní API, zastaralý software, nezabezpečený přenos a ukládání dat a v neposlední řadě slabá či žádná fyzická bezpečnost.

Řešení k těmto zranitelnostem může představovat PKI a digitální certifikáty, zabezpečení sítě, API a proti fyzickému vniknutí.

Public Key Infrastructure a digitální certifikáty

Jednou z nejúčinnějších a nejběžnějších forem zabezpečení IoT zařízení v současnosti je využívání PKI (Public Key Infrastructure). Digitální certifikáty využívají asymetrickou kryptografii, jsou skvělé pro určování identity zařízení, autentizaci a šifrování. Minimalizuje podvody ověřováním identity osob přes internet, zajistí soukromí zpráv, integritu elektronických komunikací a nepopiratelnost transakcí. Tím pádem bude menší riziko, že by si to během přenosu někdo přečetl, sníží se šance pro pozměnění nebo neoprávněné manipulace během přenosu a platnou elektronickou transakci [11].

Používají se dva typy klíčů - veřejný a soukromý. K soukromému klíči má přístup pouze vlastník digitálního certifikátu a může si vybrat, kam se veřejný klíč dostane. Certifikát funguje pro předání veřejného klíče uživatelům, které vlastních chce mít. Tyto dva klíče musí spolupracovat, jelikož soubor, který je zašifrován soukromým klíčem, lze dešifrovat pouze veřejným klíčem a naopak [12, 13].

PKI je dobré pro aplikace s požadavkem na vysoké zabezpečení. Díky digitálnímu podepisování spolu s veřejnými a soukromými kryptografickými klíči poskytuje PKI důvěru, kterou lze použít k zabezpečení různých aplikací. Digitální certifikáty prokazují integritu a identifikaci obou stran. Pomáhají ověřit, že konkrétní veřejný klíč patří určité entitě. Pokud byl certifikát vydán zdrojem, který server zná a kterému důvěřuje, server přijme certifikát jako důkaz identity [12, 13].

Zabezpečení sítě

Je nutné brát zřetel na fakt, že nejen oprávněným uživatelům internet dává vzdálený přístup k IoT zařízením, ale také útočníkům, proto je důležité myslet na síťové zabezpečení. Klíčovou roli hraje zabezpečení portů, konkrétně jejich přesměrování a neotvírání těch nepotřebných. Dále je důležité, aby autentizace byla silná a změněná a využívali se komunikační protokoly, které jsou řádně zabezpečené. Podstatné je zajistit aktuálnost softwaru, používání antimalware, firewall a blokování neautorizovaných IP adres [14].

Zabezpečení API

API (Application Programming Interface) je softwarový prostředník, který umožňuje komunikaci mezi dvěma aplikacemi. Spojuje informace a následně je vyhodnotí. Jestliže je rozhraní API narušené, může uniknout velký počet dat.

Jedním způsobem, jak chránit API je použitím autentizace a autorizace. Většinou se nevynucuje autentizace u soukromých rozhraní API, která jsou určená pro interní užití nebo využívá autentizace, která není dostatečně silná. Jestliže se jedná o větší organizace, tak ti můžou vynechat autorizaci, to znamená, že přístup k datům má prakticky kdokoliv [15].

Druhým způsobem, jak chránit svoje API je využití principem nejmenšího privilegia. Subjektům jako jsou uživatelé, procesy, programům, systémům aj. je udělené pouze minimální nezbytný přístup k provedení stanovené funkce [15].

Dalšími řešení pro ochranu API jsou – šifrování dat pomocí TLS, ukazování určitých dat, které jsou nezbytné pro plnění funkcí, používání omezených rychlostí, tj. maximum zpracování požadavků za den pro zabránění útokům jako jsou Denial of service aj. [15].

Zabezpečení proti fyzickému vniknutí

Zabezpečení bezobslužných fyzických zařízení představuje důležitý úkol v IoT. Zařízení IoT jsou nasazena v rozptýlených a vzdálených prostředích — nejsou držena v žádném kontrolovaném prostředí, ale vystavena v terénu, aby mohla provádět své operace. Útočník může narušit služby nabízené zařízeními IoT získáním přístupu a manipulací s fyzickou vrstvou. Takové akce by mohly například zabránit senzorům v detekci rizik, jako je požár, záplava a neočekávaný pohyb [12].

Pro zabezpečení takových to zařízení je klíčové, aby zařízení, která jsou v terénu a tedy potenciálně přístupná třetím osobám, dokázala rozpoznat manipulaci se zařízením, jako je například funkce detekce narušení perimetru či využití akcelerometru, díky kterému zařízení pozná přerušení spoje nebo neoprávněný přístup.

Zařízení se následně deaktivuje, zašifruje, zničí, či pošle na e-mailovou adresu informace a to kdykoli, je s ním neoprávněně manipulováno, např. otevírání krytu, zničení připojených kabelů, detekci otřesu aj. [19, 24]

2 Zabezpečení dat na platformě Raspberry Pi

Tato kapitola se bude zabývat platformy Raspberry Pi a jeho zabezpečení. Předchozí kapitola se zabývala využití IoT a hrozeb, které mohou nastat. V této kapitole se bude zabývat zabezpečení těchto zařízení, aby nebyla žádná citlivá data odcizena a útočník je nemohl využít ve svůj prospěch. Nejprve bude představena platforma, která je zabezpečována, následně produkt Zymkey 4i, zabezpečovací modul od společnosti Zymbit a nakonec Micro SD karta Swissbit PS-45u od společnosti Swissbit.

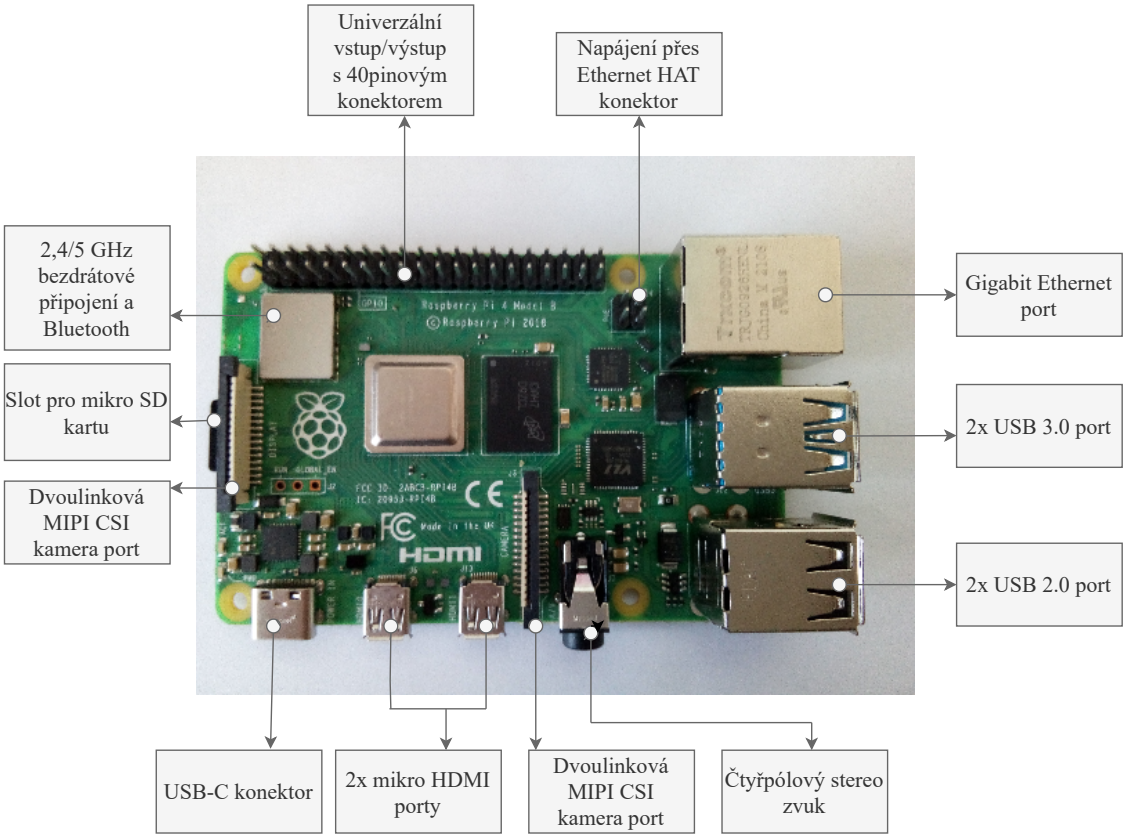
2.1 Platforma Raspberry Pi

Raspberry Pi je název pro sérii jednodeskových počítačů, kterou roku 2012 uvedla na trh nadace Raspberry Pi Foundation. Za tuto dobu vzniklo několik novějších generací a modelů Raspberry Pi. Zatím jsou vytvořeny celkem čtyři generace, které jsou si podobné a splňují stejný cíl, tedy malý rozměr, výkonnost a malá finanční náročnost. Obecně platí pravidlo, že Model A a Model B existují skoro u všech generací. Model A je levnější variantou, ale také horší, protože má sníženou paměť RAM a méně portů. Všechny generace kromě druhé generace mají navíc modely s označením „plus“. Tyto modely jsou lepší, jelikož jsou dokonalejší verzí předešlých modelů. Raspberry Pi Zero je ještě menší a levnější verzí. O dva roky později vyrobili vývojáři podobné zařízení k Raspberry Pi Zero, tj. Raspberry Pi Zero W, kde W značí wireless tedy Wi-Fi připojení, které je vhodné pro IoT zařízení protože umožňuje bezdrátové připojení k internetu. Nejnovější generací je Raspberry Pi 4. V roce 2021 vznikl již Raspberry Pi 400, který slouží jako miniaturní počítač zabudovaný v klávesnici [17].

Cílem Raspberry Pi Foundation je vzdělávat lidí v oblasti výpočetní techniky a vytváření snadnějšího přístupu k počítačovému vzdělávání. V rámci tohoto cíle je možno dělat na počítači vše, co je potřebné znát v každodenním životě – např. přehrávání videa, přístup k internetu, hraní her, vytváření tabulek a programování. Raspberry Pi také umožňuje vytvořit si vlastní chytrou domácnost, webový server, ovládat robota, vybudovat bezpečnostní systém a mnoho dalšího [16].

V této práci je použitý Raspberry Pi 4 model B, kterou je také nejnovější generací od společnosti Raspberry Pi Foundation. Rychlost procesoru byla zvýšena stejně jako multimediální výkon, paměť a konektivita oproti minulých generací, ale velikost zařízení je stále stejná tedy 85x56 milimetrů. Nejnovější model byl vydán v roce 2020. Nachází se zde čtyřjádrový procesor Cortex-A72 (ARM v8) naladěný na 1,5 GHz, grafická karta využívá Broadcom VideoCore VI s čipem Broadcom BCM2711.

Oproti minulých generací si lze vybrat kapacitu paměti RAM 1 GB, 2 GB nebo 4 GB. Nalezneme zde bezdrátové připojení IEEE 802.11ac v rozsahu 2,4 GHz i 5,0 GHz, Bluetooth 5.0 BLE, ale i drátový nelimitovaný Gigabit Ethernet. K připojení vstupních zařízení můžeme využít dva USB 3.0 porty anebo dva USB 2.0 porty. Dále je zde standardní 40pinový GPIO header, dvoupásmový MIPI DSI display port, dvoupásmový MIPI CSI port kamery, čtyřpólový stereo audio a kompozitní video port, 2 micro-HDMI porty k zapojení obrazovky, kde umožněno rozlišení obrazu až až 4K-60p. Napájení probíhá přes USB-C konektor [18]. Podobu tohoto modelu a popis jeho komponent zobrazuje obrázek 2.1.



Obr. 2.1: Rozhraní modelu Raspberry Pi 4.

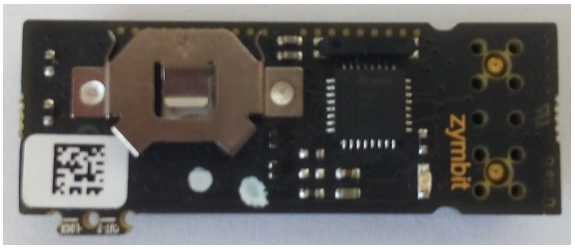
2.2 Zymbit

Zymbit je společnost, která chrání bezobslužný majetek před kybernetickým a fyzickým zneužitím. Společnost poskytuje autonomní zabezpečení postavené na změřené identitě systému, šifrování a podepisování dat, digitální peněžence a na dalších kryptografických službách. Široká škála možností je obsažena v modulu citlivém na neo-

právněnou manipulaci se zabezpečeným softwarovým API. Celkem nabízejí 3 různá zařízení – model Zymkey 4i, HSM4 a HSM6.

2.2.1 Zymkey 4i

Zymkey 4i se zapojuje přímo do Raspberry Pi GPIO pomocí 10pinového konektoru. K dispozici jsou API jsou pro programovací jazyky Python, C a C++. Dokáže generovat zabezpečené páry klíčů – tedy veřejné klíče k šifrování a soukromé klíče k dešifrování. Soukromý klíč je chráněný. Úložiště klíčů je zabezpečené odolným prvkem proti neoprávněné manipulaci. Má tři páry soukromého a veřejného klíče. Využívá kryptografické prvky TRNG (Hardwarový generátor náhodných čísel), který slouží například pro vytvoření náhodných kryptografických klíčů potřebný k šifrování a podepisování dat. ECDSA (Protokol digitálního podpisu s využitím eliptických křivek), který se používá k digitálním podpisům, AES-256 (Advanced Encryption Standard s 256bitovým klíčem) využívaná šifra pro šifrování dat a ECC NIST P-256. Při fyzické manipulaci má dva detekční obvody, akcelerometr otřesu i orientační sensor a monitor napájecí lišty. Dokáže zašifrovat kořenový souborový systém pomocí nástroje `dm-crypt` a klíčového manažera LUKS (Linux Unified Key Setup). Šifruje datové BLOB (Binary Large Object) díky funkci `zblock`. Šifruje data pomocí integrace OpenSSL. Lze snadno integrovat s Amazon Web Services IoT služby registrace certifikátů v čase [21]. Na obrázku 2.2, lze vidět modul Zymkey 4i.



Obr. 2.2: Model Zymkey 4i.

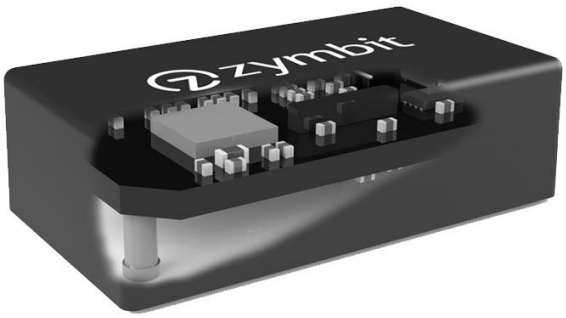
2.2.2 HSM4

HSM4 je rozšířeným modelem Zymkey 4i s 30pinovým konektorem, jenž je zapouzdřen do zabezpečeného modulu s jediným skrytým konektorem. Je vhodnější do velkoobjemových aplikací. Podporuje operační systémy Raspberry Pi OS a Ubuntu.

2.2.3 HSM6

HSM6 je nejnovější model od Zymbit. Funguje jako hardwarová zabezpečená peněženka. Podporuje blockchainy a krypto aplikace běžící na jednodeskových počítačích. Jeho součástí je stejně jako u HSM4 30pinový konektor.

HSM6 je současně rozšířený HSM4. Lze zde zadat až 512 párů klíčů a 128 cizích veřejných klíčů, který si uživatel sám zvolí. Dále je zde BIP 32/39/44 HD peněženka nebo Hierarchická deterministická peněženka, digitální peněženka, která je běžně používaná k ukládání klíčů pro držitele kryptoměn. ECC KOBLITZ P-256 (secp256k1) odkazuje na parametry eliptické křivky používané v kryptografii, její součástí je veřejný klíč Bitcoinu, který je definován standardem pro efektivní kryptografii a využívá další kryptografické prvky, například dokáže vnímat teplotu bezpečnostního modulu [21]. Lze vidět na obrázku 2.3 společně s HSM4.



Obr. 2.3: Model HSM4 a HSM6 [22].

2.2.4 Funkce Zymkey 4i

Tato kapitole se bude zabývat detailnější implementací funkcionality v Zymkey 4i, tedy co všechno dokáže tento modul nabídnout.

Hlavními funkcemi modulu, které hrají důležitou roli v kryptografických operací a které zde budou představeny a zároveň v praktické části této práci předvedeny, jsou funkce `lock` a `unlock`, tj. šifrování a dešifrování. Dále bude představena pro případ fyzického vniknutí funkce `set_tap_sensitivity`, díky které lze nastavit citlivost zařízení a odpovídající reakce na vniknutí, `wait_for_tap`, jenž čeká na události dotyku a `get_accelerometer_data` sloužící pro získání aktuálních dat akcelometru. Perimetr slouží k přerušení obvodu. Tudíž případná manipulace či násilné ustřihnutí obvodu perimetru bude vést k zapnutí a upozornění zařízení. Nejdůležitější funkcí pro fyzické vniknutí je `set_perimeter_event_actions`, který

nastaví akci narušení obvodu. Funkce `wait_for_perimeter_event` čeká na narušení perimetru, `get_perimeter_detect_info` získá informace o detekci obvodu a `clear_perimeter_detect_info` vymaže informace o detekci obvodu. Detailnější popis funkce je uveden v následujících podkapitolách.

Lze zde také najít funkce jako je „sign“, díky kterému je vygenerován podpis pomocí soukromého klíče ECDSA Zymkey využívající hašovácí funcke SHA-256 (Secure Hash Algorithm). Funkce „verify“ a `verify_digest` dokáží ověřit data pomocí podpisu. K vytvoření souboru s veřejným klíčem ECDSA ve formátu PEM lze použít funkce `create_ecdsa_public_key_file`. Tato metoda Vám umožní generovat žádosti o podpis certifikátu. Díky `store_foreign_public_key`, dokáže ukládat cizí veřejný klíč do cizího veřejného klíče Zymkey a mnoho dalšího.

Funkce lock

`Lock` funkce slouží k uzamknutí prostého textu dat, kde se zašifruje a podepíše tento blok. Jednosměrný klíč je určen k uzamčení dat pouze na místním hostitelském počítači. Data šifrovaná pomocí tohoto klíče nelze exportovat a dešifrovat nikam jinam. Do funkce lze zadat až tři parametry, jeden povinný a dva volitelné. Prvním parametrem jsou data, která se budou uzamykat. Druhým je cíl šifrovaného textu. Posledním parametrem je `encryption_key`, který určuje, jaký klíč bude použit k uzamčení dat. Hodnota „zymbit“ je výchozí a určuje, že Zymkey 4i bude používat jednosměrný klíč. Pokud není zadán žádný cíl vrací data jako bytearray v opačném případě je `None`. Příklad využití funkce je uveden ve výpisu 2.1.

Výpis 2.1: Příklad použití funkce lock.

```
import zymkey
...
zymkey.client.lock(src = f"tempFile/hexaData{index}.txt",
dst = encryptedFile,
encryption_key = "zymmkey")
```

Funkce unlock

Funkce `unlock` funguje na stejné bázi jako `lock`, a to pouze s opačným provedením, tedy ověřuje podpis uzamčeného objektu a dešifruje přidružená data šifrovaného textu. Jednosměrný klíč je určen k uzamčení dat pouze na místním hostitelském počítači. Data zašifrovaná pomocí tohoto klíče nelze exportovat a dešifrovat nikam jinam, stejně tomu je u funkce `lock`. Parametry jsou také podobné, přibyla zde jedna funkce navíc, a tou je `raise_exception` typu bool. První parametr ověřuje a dešifruje šifrovaný text, druhý parametr zadává cíl dešifrovaných dat na prostý text

– klíč, který bude použit k uzamčení a poslední parametr k vyvolání výjimky, pokud selže ověření podpisu zamčeného objektu. Hodnotu vrací stejně jako u metody `lock`. Příklad využití funkce je uveden ve výpisu 2.2.

Výpis 2.2: Příklad použití funkce unlock.

```
import zymkey
...
zymkey.client.unlock(src = encryptedFile,
dst = f"tempFile/decryption{index}.txt",
encryption_key = "zymkey",
raise_exception = True)
```

Funkce set_tap_sensitivity

Metoda umožňuje nastavit citlivost dotyku. Každá osa může být nastavena buď individuálně nebo mohou být všechny osy nastaveny naráz. Jsou zde vyžadovány dva parametry, tj. osa a citlivost. První parametr je osa, která může být definována jako all, tedy všechny osy se zadanou stejnou hodnotou nebo osa x, y, z, jež lze nastavit individuálně. U druhého parametru je citlivost vyjádřena v procentech. Je-li nastavena na 0 %, detekce je vypnuta a 100 % znamená, že detekce je dána na maximální hodnotě, tudíž citlivost je velmi vysoká. Příklad použití funkce je uveden ve výpisu 2.3.

Výpis 2.3: Příklad použití funkce setTapSensitivity.

```
import zymkey
...
zymkey.client.set_tap_sensitivity(axis = "all",
pct = 50.0)
```

Funkce wait_for_tap

Tato funkce čeká na pohyb či dotyk, v dalším kroku blokuje volající vlákno, pokud není voláno s časovým limitem nula. Je zde jeden jediný parametr, díky kterému je možnost zadat maximální doba v milisekundách, a to pro čekání na příchod události dotyku. Příklad použití funkce je uveden ve výpisu 2.4.

Výpis 2.4: Příklad použití funkce waitForTap.

```
import zymkey
...
zymkey.client.wait_for_tap(timeout_ms = 1000)
```

Funkce get_accelerometer_dat

Tato metoda získá nejnovější data akcelerometru v jednotkách g (gravitační přetížení tělesa) plus směr závitu na osu. Je zde mnoho návratů, např. řada hodnot akcelometru v jednotkách g, index pole 0 = osa x, 1 = osa y, 2 = osa z, hodnota -1 znamená, že událost dotyku byla detekována. Příklad použití funkce je uveden ve výpisu 2.5.

Výpis 2.5: Příklad použití funkce getAccelerometerDat.

```
import zymkey
...
a[0], a[1], a[2] = zymkey.client.get_accelerometer
_data()
```

Výpis 2.6: Běh funkce getAccelerometerDat

```
Nothing is going on 2021-12-06 time 22:06:55.042828
Nothing is going on 2021-12-06 time 22:06:56.155376
Touch detected
Accelerometer was detected 2021-12-06 time 22:06:56.65205
X:
g-force = -0.0039072041399776936,
tap direction = 0
Y:
g-force = -1.437851071357727,
tap direction = -1
Z:
g-force = -1.5980464220046997,
tap direction = 0
Choose what do you want to do with the device 1 = warning
2 = self-destruction
Email was sended
Nothing is going on 2021-12-06 time 22:06:58.043716
```

Výpis 2.6, ukazuje výstup funkce `get_accelometer_data`. Akcelerometr se aktualizuje každou jednu sekundu a říká jaký stav právě probíhá. První dvě sekundy se nic neděje a následně zaznamená pohyb. Vypíše osu x,y,z v jednotkách g a směr dotyku na zařízení. Následně se vybírá, zda chcete, aby přišlo upozornění na e-mail nebo provézt sebedesktrukci. V tomto případě byl vybrán upozornění na mail a dále už opět nebyl záznam pohybu.

Funkce set_perimeter_event_actions

Funkce určuje akci, která se má provést, jestliže dojde k narušení obvodu. Možné akce jsou pak informovat hostitele nebo sebedestrukce Zymkey. Má tři parametry – jednou je kanál (0 nebo 1) na, kterým kanále bude použitá daná akce, druhým je upozornění pro narušení perimetru a poslední definitivní akcí je zničení samotného Zymkey. Příklad využití funkce je uveden ve výpisu 2.7.

Výpis 2.7: Příklad použití funkce setPerimeterEventActions.

```
import zymkey
...
zymkey.client.set_perimeter_event_actions(channel = 1,
action_notify = True,
action_self_destruct = False)
```

Funkce wait_for_perimeter_event

`Wait_for_perimeter_event` obsahuje podobné funkce jak u akcelometru tj, `wait_for_tap`. Čeká na jakékoliv narušení perimetru. Tato funkce blokuje volající vlákno, pokud ovšem není voláno s časovým limitem nula. Parametr je pouze jeden, díky jemuž lze zadat maximální dobu v milisekundách pro čekání na příchod události klepnutí. Příklad využití funkce je uveden ve výpisu 2.8.

Výpis 2.8: Příklad použití funkce waitForPerimeterEvent.

```
import zymkey
...
zymkey.client.wait_for_perimeter_event(timeout_ms = 1000)
```

Funkce get_perimeter_detect_info

V rámci funkce `get_perimeter_detect_info` se získá aktuální informace o detekci obvodu. Tato funkce získá časové razítko první události detekce perimetru pro daný kanál. Index odpovídá kanálu specifikovanému v `set_perimeter_event_actions`. Poté navrací hodnotu podle časových značek pro každý kanál po první detekci. Příklad využití funkce je uveden ve výpisu 2.9.

Výpis 2.9: Příklad použití funkce getPerimeterDetectInfo.

```
import zymkey
...
zymkey.client.get_perimeter_detect_info()
```

Funkce clear_perimeter_detect_info

Metoda `clear_perimeter_detect_info` vymaže všechny informace o detekci obvodu a znovu aktivuje všechny kanály pro detekci obvodu. Příklad využití funkce je uveden ve výpisu 2.10.

Výpis 2.10: Příklad použití funkce clearPerimeterDetectInfo.

```
import zymkey
...
zymkey.client.clear_perimeter_detect_info()
```

2.2.5 Pracovní režimy Zymkey 4i

V Zymkey 4i lze nastavit dva režimy. Jedním je vývojářský režim, který umožnuje jakkoliv upravovat, přidávat nebo odstraňovat funkce či změnit hostitelské zařízení a druhým režimem je produkční režim, který je nenávratným krokem, to znamená, že jakmile dojde k jeho aktivaci, nelze už nijak modifikovat a změnit hostitelské zařízení.

Vývojářský režim

Termín vývojářský režim zahrnuje režím, díky kterému se může vývojář ujistit, že hostitel má všechny nezbytné předpoklady pro propojení se Zymkey 4i a také, že bude schopen spustit softwarovou aplikaci. Stále však může pracovat a ovládat svoji softwarovou aplikaci. Jakýkoliv krok, který udělá, je návratný a smí ho pozměnit. Také je možno kdykoliv změnit hostitelské zařízení, aniž by se nějaké údaje ztratily či zničily. Tento krok je velmi vhodný pro vyvíjení aplikací, protože žádná data nebudou poškozena nebo ztracena a nabízí se možnost jakkoliv měnit hostitelské zařízení [24].

Produkční režim

Po nastavení produkčního režimu bude konkrétní zařízení Zymkey 4i trvale uzamčen pro konkrétní hostitelské zařízení. Neexistuje zde žádný tovární reset, hlavní klíč ani jiné formy obnovy. Je velmi důležité dbát na postup, který je popsán v dokumentaci API od Zymkey 4i pro jednotlivé funkce, protože pokud nebude dodržen postup například u perimetru, může nastat situace, že zařízení nebude fungovat, tak jak bylo požadováno. Pokud bude provedeno zásadní vylepšení (upgrade) distribuce, Zymbit nemůže zaručit jeho správný provoz. Další nespornou výhodou je fakt, že je vhodný do praktického nasazení, protože jej nelze už nijak upravovat a změnit na jiné

hostitelské zařízení. Z toho plyne, že je zásadní, aby aplikace byla řádně testována s hostitelským zařízením ještě předtím, než bude použit produkční režim [24].

2.3 Vazba a autentizace Zymkey 4i

Zabezpečení autentizace začíná až při přiřazení jedinečné a nezměnitelné identity (identifikační číslo) každému zařízení, která se používá k ověření následných interakcí se zařízením. Zymkey generuje jedinečné ID (identifikace) zařízení měřením určitých atributů konkrétního hostitele a poté zkombinováním tohoto měření s jedinečným ID tohoto konkrétního Zymkey. Proces kombinování těchto identifikátorů využívá kryptografickou funkci a tento proces se obecně nazývá vazba [26].

Pro vazbu těchto zařízení je nutné, aby byl aktivován Produkční režim, protože u Vývojářského režimu je vazba dočasná a tím pádem zranitelnější, kvůli tomu že může být vytáhnuta SD karta nebo odebrán modul z Raspberry Pi [26].

2.4 Swissbit

Společnost Swissbit AG byla založena v roce 2001 a její pobočky se nachází jak v Asii, tak i na americkém a evropském kontinentu. Je předním evropským výrobcem úložných, bezpečnostních a vestavěných IoT řešení. Tato znalost umožňuje zákazníkům spolehlivě ukládat a chránit data v různých odvětvích, tj. v průmyslovém, NetCom, automobilovém, lékařském, ve fiskálních aplikacích a v neposlední řadě Internetu věcí. Swissbit vyvíjí a vyrábí průmyslové, úložné a bezpečnostní produkty s dlouhodobou dostupností, vysokou spolehlivostí a vlastní optimalizací [25]. Možnosti zabezpečení Swissbit přidávají novou úroveň ochrany uložených dat, nebo dokonce zabezpečení celého systému tím, že nabízejí skutečně bezpečné úložiště klíčů prostřednictvím upgradovatelného a vyměnitelného hardwarového zařízení [27].

Jelikož Internet věcí je rychle rostoucí položka na trhu pro vestavěné a průmyslové systémy s nedostatečným fyzickým zabezpečením, jsou zde produkty od společnosti Swissbit, který problém nedostatečného zabezpečení řeší. Swissbit poskytuje různé typy úložných řešení pro systémy IoT, jenž umožňují spolehlivý, stabilní a vysoce výkonný provoz s dlouhou životností [28].

Aby bezpečnostní produkty fungovaly jak mají, je zapotřebí dbát na spolehlivost a teplotní rozsah, tj. velké výkyvy teplot, náhlé výpadky proudu či neovlivnitelné jevy prostředí. Musí být zaručena vysoká odolnost produktů, malé rozměry pro umožnění kompaktního systému a zabezpečení lokálního úložiště před neoprávněnou manipulací a krádeží. Veškeré tyto problémy řeší produkty od společnosti Swissbit [28].

Produkty Swissbit PCIe a SATA poskytují správnou kapacitu úložiště, životnost a rozsah provozních teplot pro venkovní i vnitřní systémy. Využívání technologie 3D NAND pro zachování vysoké spolehlivosti a výdrže. Také používají SD a micro SD karty, eMMC disky a USB produkty s vysokou výdrží. Samotné produkty chrání pomocí PIN kódu, USB flash disky s řízením přístupu a šifrováním [28].

2.4.1 Micro SD karta Swissbit PS-45u

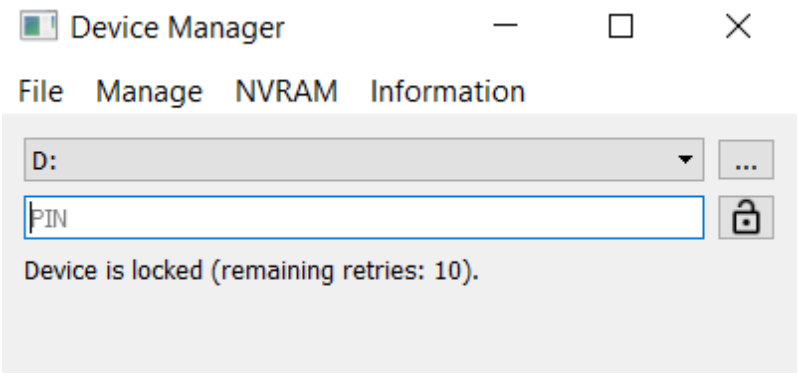
Paměťová karta PS-45u nabízí velikost paměti 8 nebo 32 GB. Je to standartní mikro SD karta s rozměry 15.0mm x 11.0mm x 0.7mm a plně kompatibilní se specifikací SD karet 2.0 a 3.0 a mikro SD karet. Rychlost sekvenčního čtení je až 17 MB/s a rychlost sekvenčního zápisu nabízí až 13 MB/s. Dokáže fungovat v teplotách rozmezí -40 °C až v 85 °C. Funguje s moduly Raspberry Pi 2, 3B+, CM3+, kde je podporovaná politika USB, PIN a NET a 4, kde chybí podpora pro USB. Pro používání autentizace pomocí USB je důležité koupit USB autentizační dongle Swissbit USB-Stick PU-50n DP 'Raspberry Edition'. K šifrování dat využívá šifrování AES-256. Do klíčových vlastností Mikro SD karty patří bezpečnostní zásady s flexibilní a konfigurovatelnou autentizací, ověřování uživatele se provádí během startovací fáze před spuštěním systému Swissbit, aby se odemkl přístup a ochrany přístupu s konfigurovatelným čítačem opakování. Zaručuje IP ochranu uzamčením mikro SD karty, ochrany proti krádeži a řízení licencí poskytnutím jedinečného ID [29].

Obr. 2.4: Mikro SD karta Swissbit PS-45u

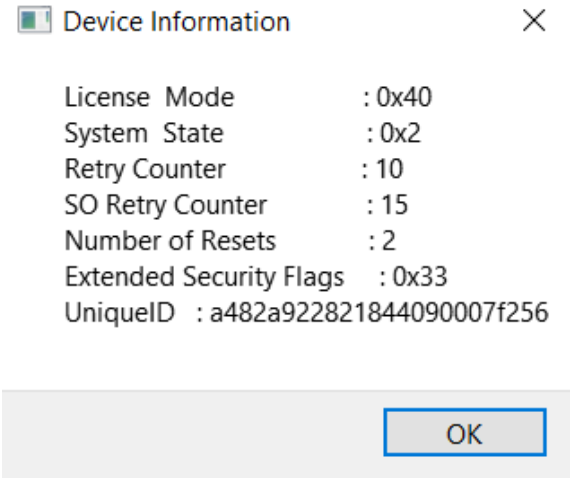
Každá politika zabezpečení má svoje výhody a nevýhody. Stejné zabezpečovací funkce používá PIN, USB a NET, tj. ochrana Know-How, licence, dat a dálkové ověření. Kde se odlišují ochrana proti krádeži, která je využitá jen u PINu a NET politice. Bezpečné bezobslužné spouštění dokáže USB tak i NET. NET politika, ale má nejsilnější ochranu, protože jediná tato politika využívá zabezpečení lokálního úložiště před neoprávněnou manipulací a uzamknutí zařízení na dálku [29].

Swissbit využívá grafické uživatelské rozhraní pro nastavení SD karty viz obrázek 2.5. Je zde **Manage**, kde se nastavuje změna PIN, aktivace či deaktivace ochrany dat, ochranné nastavení aj. V NVRAM (Non-Volatile Random-Access Memory) lze

najít konfiguraci a nastavení jednotlivých politik, které se dají ručně zadat. Information, česky informace, najdeme veškeré potřebné informace ohledně zařízení. Nejzajímavější informace lze najít pod uniqueID neboli unikátním ID, které má každá Swissbit jedinečné, a které funguje jako autentizační číslo pro NET server [29]. Viz obrázek 2.6.



Obr. 2.5: Správce zařízení SD karty PS-45u



Obr. 2.6: Informace o zařízení SD karty PS-45u

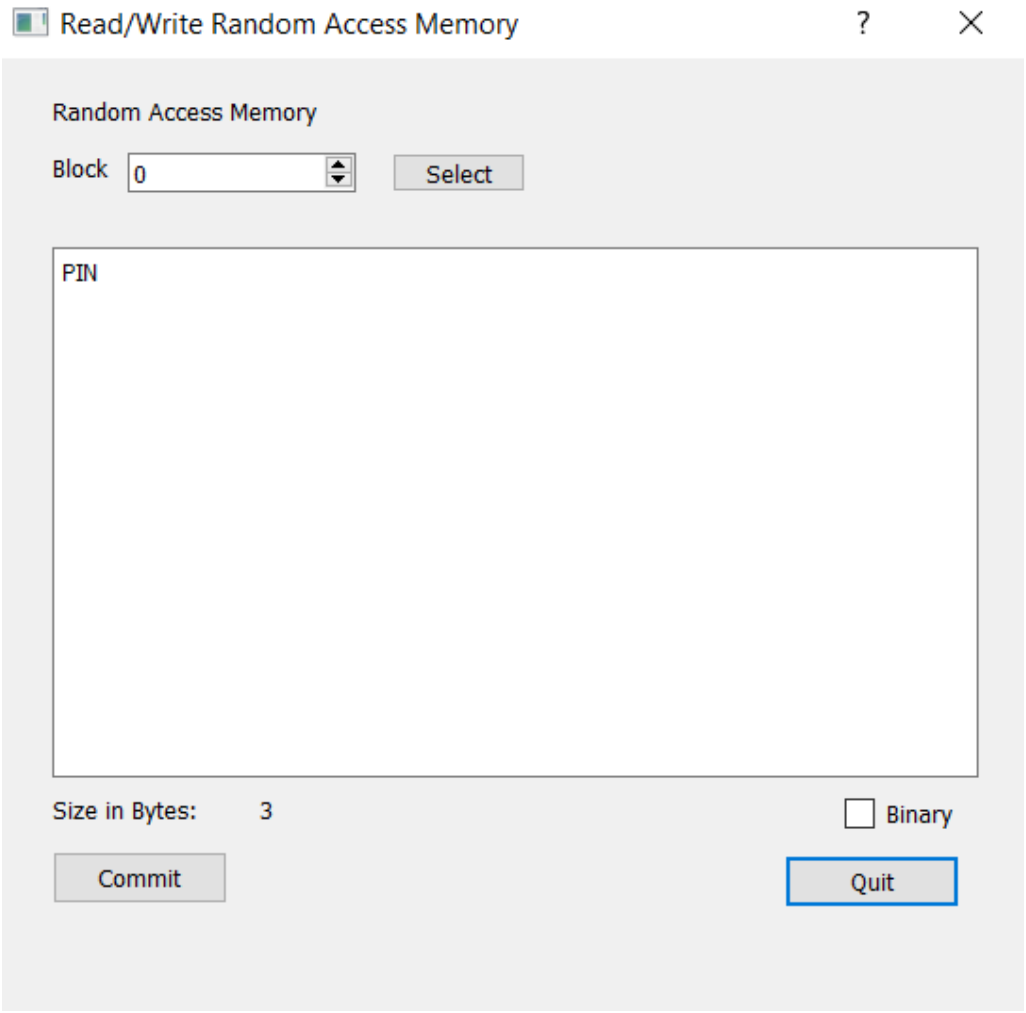
Všeobecně na SD kartu lze nastavit ochranu na více oddílů disku, zabezpečit pomocí PIN vstup do kořene SDK (Software development kit), podporuje fast wipe a při přihlášení je umožněno tzv. Soft reset, kdy se restartuje zařízení za účelem vymazání jeho vnitřní paměti od spuštěných programů. Dále je možné vybrat, jaké oprávnění je umožněno (čtení, psaní) pro daného, konkrétního uživatele [29].

2.4.2 Autentizace na Swissbit SD kartě

Uživatelé mohou implementovat bezpečnostní zásady třemi různými způsoby: pomocí zadání PIN kódu, díky USB nebo přes Net Policy Server. Autentizace je chráněna tokenem hašovací funkce SHA-256. To vše lze nastavit jednoduše pomocí GUI (grafické uživatelské rozhraní), které lze stáhnout z jejich oficiálních stránek.

Autentizace pomocí PIN kódu

Autentizace pomocí PIN (personal identification number) kódu funguje na bázi několika místního ověřovacího kódu, které zadá uživatel, aby odemkl kartu pro další proces spuštění. V liště NVRAM a kolonce Read/Write Random Access Memory se zadá zabezpečovací politika pomocí PIN kódu. Tuto operaci znázorňuje obrázek 2.7.

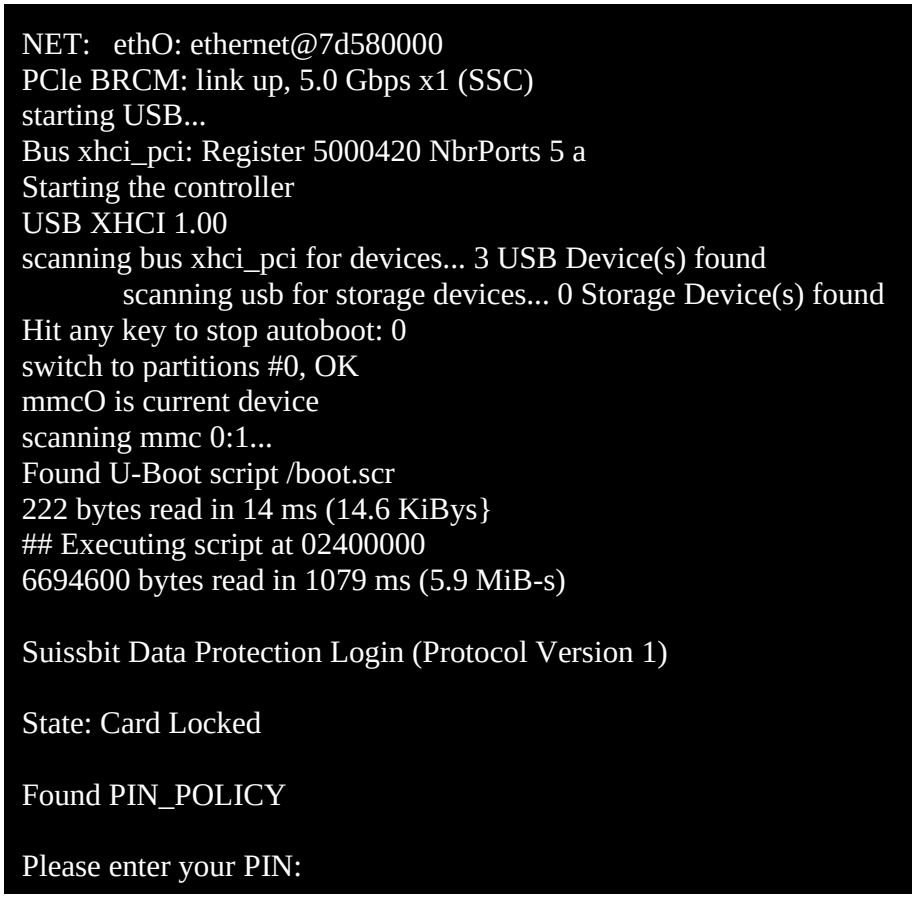


Obr. 2.7: Nastavení PIN kódu na SD kartě PS-45u.

Následně po vytvoření autentizace je zapotřebí aktivovat toto zabezpečení, kde je

nutné zadat PIN a officer PIN. Dále obsahuje počítadlo, které nastaví počet pokusů, při kterých bude umožněno zopakovat heslo. Jestliže heslo nebude úspěšné ani po maximálním počtu povolených pokusů uživatelem, tak se SD karta blokuje a lze se pouze autentizovat pomocí officera hesla.

Jak vypadá autentizace pomocí PIN kódu na SD kartě, lze vidět na obrázku 2.8. V tento moment je zařízení uzamčené a lze pouze psát, protože veškeré funkce jsou zablokovány do doby zadání správného hesla.

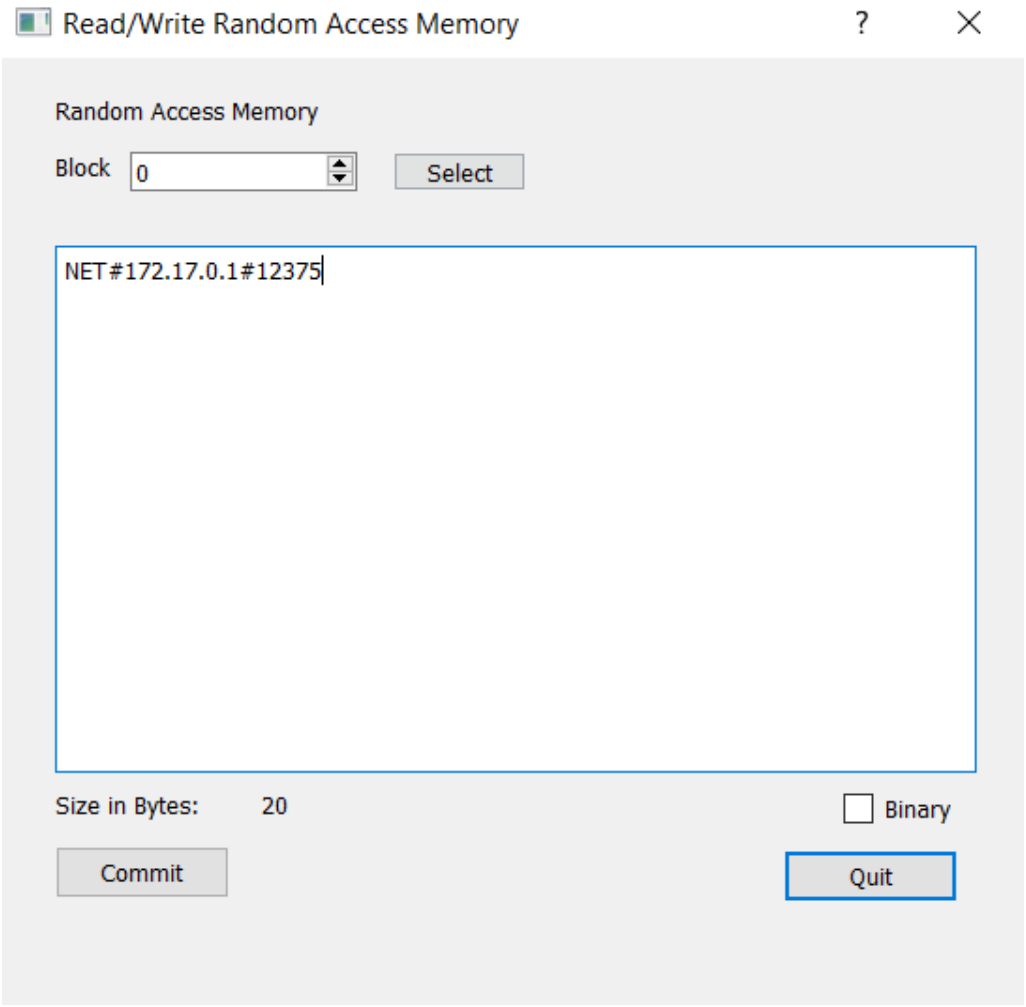


Obr. 2.8: autentizace pomocí PIN kódu na SD kartě PS-45u.

Autetizace pomocí NET serveru

NET policy server znamená, že během procesu spouštění bude U-Boot získávat ověřovací informace z autentizačního serveru v síti. Je zapotřebí, aby server byl linuxová distribuce, v tomto případě se využívá virtuální počítač Ubuntu. Stejně jako i PIN lze nastavit v liště NVRAM a kolonce Read/Write Random Access Memory, zabezpečovací politiku pro NET server `NET\#<IPadresa>\#<port>`, kde IP adresa

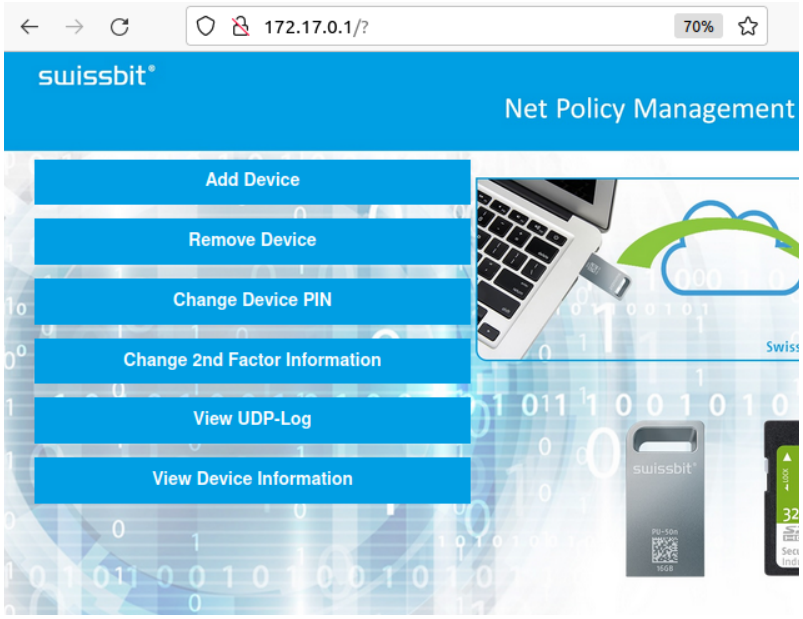
je název autentizačního serveru a port je výchozí nastaven na 12375. Tuto operaci znázorňuje obrázek 2.9.



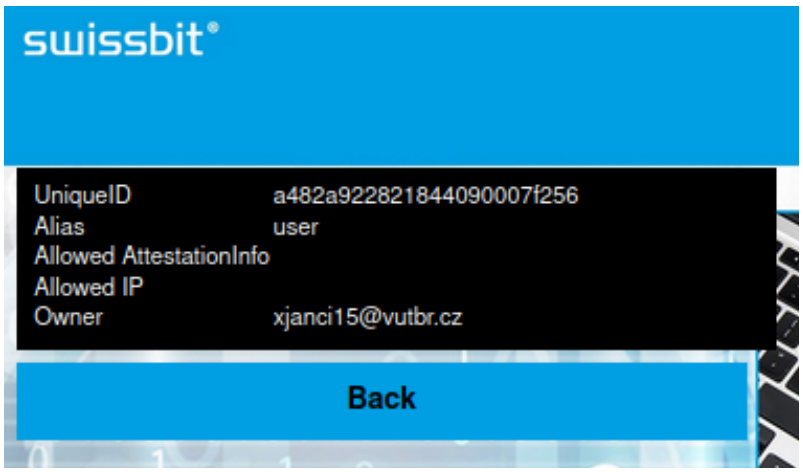
Obr. 2.9: Nastavení NET serveru na SD kartě PS-45u.

Veškeré ovládání probíhá na straně serveru v tomto případě na straně 172.17.0.1, kde může uživatel vidět detailněji informace, dále přidat, odebrat a změnit PIN zařízení, využít dvoufázovou autentizaci a podívat se do UDP-logu viz obrázek 2.10. Na obrázku 2.11 lze vidět informace o daném zařízením. E-mail slouží k odeslání varování, pokud dojde k neoprávněné manipulaci se zařízením a také slouží k dvoufázové autentizaci.

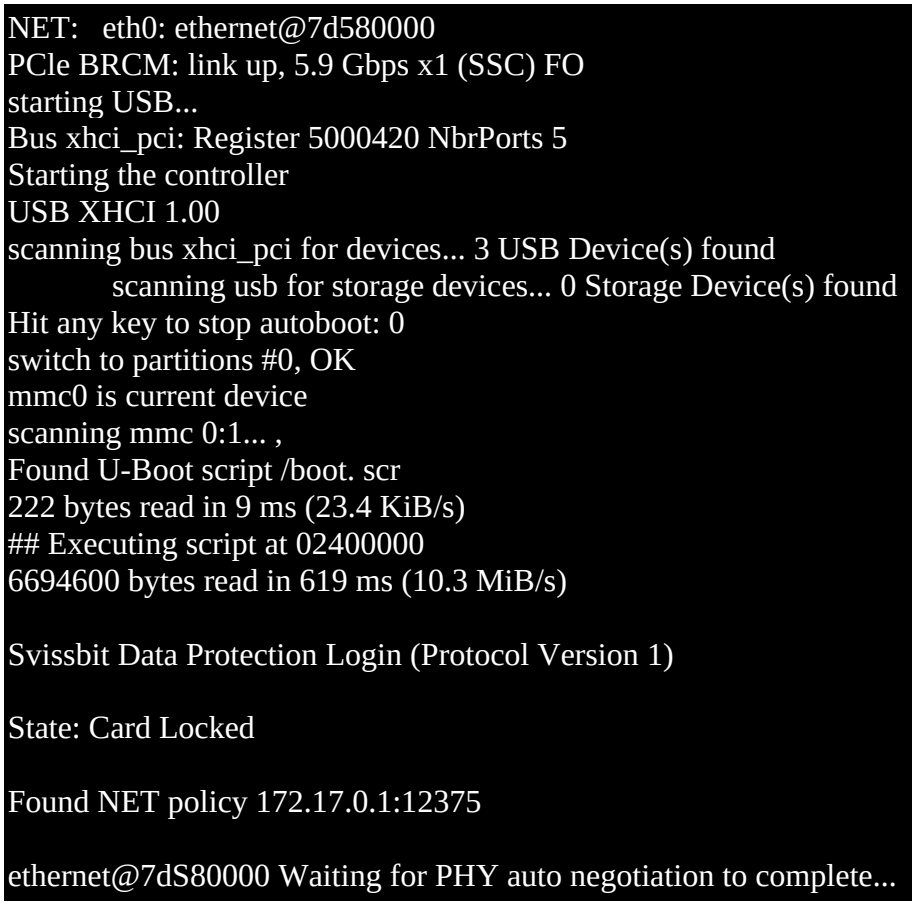
Na obrázku 2.12 se poukazuje, jak vypadá výpis autentizace pomocí NET serveru na SD kartě PS-45u. V tento moment je zařízení uzamčené a čeká na ověření serveru.



Obr. 2.10: Nastavení na web serveru pro SD kartu PS-45u.



Obr. 2.11: Informace o uživateli.



Obr. 2.12: Autentizace pomocí NET serveru na SD kartě PS-45u.

3 Návrh fyzického bezpečnostního modulu

V této části práci budou srovnány existující bezpečnostní moduly z předchozí kapitoly. Dále budou popsány definice požadavků pro uchování bezpečnosti dat na SD kartě v Raspberry Pi 4 a sestrojení vlastního návrhu fyzického bezpečnostního modulu. Je nutno podotknout, že Raspberry Pi 4 musí být chráněno proti fyzickému vniknutí, tj. veškerá data, která jsou uložená na disku, musí být bezpečně uchována na SD kartě, aby všechen důležitý obsah nemohl být odcizen.

3.1 Porovnání existujících modulů

SD karta Swissbit a Zymkey 4i chrání zapsaná data na SD kartě pomocí vlastních funkcí a různých autentizačních prvků. Ochrana dat na SD kartě Swissbit probíhá prostřednictvím autentizace při zavádění operačního systému pomocí PIN, USB nebo Net Policy Server, které se nastavují v jejich GUI aplikaci. Díky tomu je možné SD kartu připojit na jiné zařízení, jelikož nemá žádnou pevnou vazbu na stejném zařízení. Zymkey 4i naopak využívá pro autentizaci jedinečný a nezměnitelný identifikátor, z toho důvodu nelze odpojit Zymkey 4i ze zařízení a znovu využít na jiné zařízení. Modul vlastní také API, pomocí kterého se dají naprogramovat funkce perimetr a akcelerometr, díky kterým lze bezprostředně zničit SD kartu nebo poslat zprávu na e-mail. Oba dva moduly využívají pro ochranu dat uživatelů symetrickou blokovou šifru, tedy šifrovací algoritmus AES s 256bitovým klíčem.

3.1.1 Nedostatky existujících modulů

Nevýhodou Zymkey 4i spočívala v jejím produkčním režimu, rozepsaném v podkapitole 2.2.5, který se musí nastavit pro absolutní ochranu. Pokud se nastaví produkční režim, tak již není možné žádných úprav modulu a nelze jej implementovat na jiná zařízení. U Swissbit se tento problém neděje, jelikož návrh SD karty dovoluje měnit zařízení. Naopak je zde problémem velká pořizovací cena – v přepočtu 2350 korun českých [30], fixní škálovatelnost paměti a zcela zde chybí detekce perimetru a akcelerometru. Naproti tomu u Zymkey 4i nezáleží na velikosti paměti SD karty vložené v zařízení. Akcelerometr a perimetr lze naprogramovat pomocí jednoduchých funkci, popsány v podkapitole 2.2.4, a pořizovací cena je nižší než Swissbit SD karta, tj. v přepočtu 1093 korun českých [31]. V obou případech platí, že pokud by neměly moduly přístup k internetu, nezjistí obsluha či uživatel, kdy došlo k odcizení karty.

3.2 Požadavky fyzického zabezpečení SD karty

Pro fyzické zabezpečení dat na SD kartě v Raspberry Pi 4 je zapotřebí zavézt autentizaci zařízení s modulem pro oprávněný přístup k datům. Byl vytvořen účinný a rychlý šifrovací algoritmus pro ochranu dat na SD kartě, byly naimplementovány senzory pro detekci akcelerace zařízení a detektory na otevření krabice a navíc došlo k eliminaci nedostatků existujících zabezpečovacích modulů.

3.3 Vlastní návrh fyzického bezpečnostního modulu

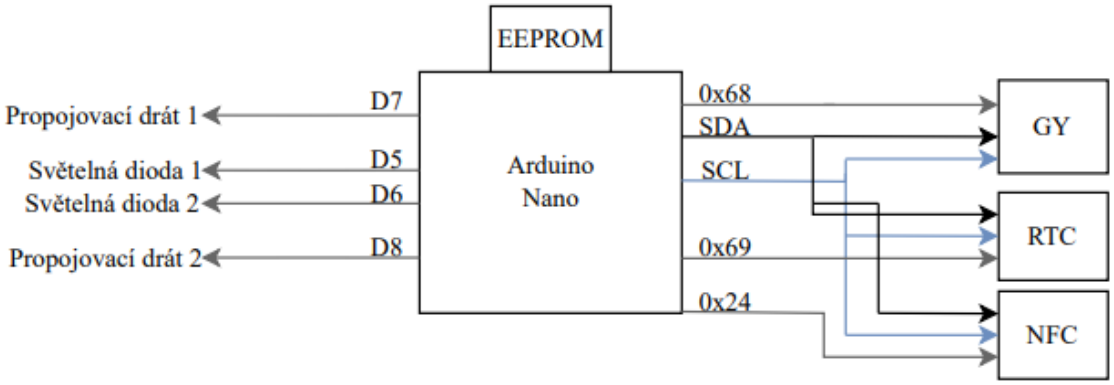
Vlastní návrh fyzického bezpečnostního modulu bude navržen tak, aby splňoval primárně požadavky pro fyzické zabezpečení SD karty. Raspberry Pi 4 bude uzavřeno v krabici a bude sbírat specifická data. Pro realizaci prototypu byl použit jednočipový počítač Arduino Nano, který poskytuje rozhraní I^2C, Inter-Integrated Circuit, sběrnici pro komunikaci na prototypové desce s ostatními komponenty. Autentizace bude probíhat pomocí uloženého klíče AES-256 v Arduino Nano. Samotná komunikace s Raspberry Pi 4 se spojí pomocí sériové komunikace s typem Arduino Nano. Podkapitoly níže se zabývají detailnějším návrhem jednotlivých kroků.

3.3.1 Prototypová deska s I^2C komunikací

Na prototypové desce budou napájeny jednotlivé komponenty pro splnění požadavků fyzického zabezpečení SD karty. Arduino Nano je Master zařízení, které zahajuje a ukončuje komunikaci, generuje hodinovou linku SCL (Serial Clock Line) pro I^2C komunikaci, která je napájena na prototypovou desku. Dalšími komponenty se stanou Slaves, které čekají na příchozí komunikaci od Master. Slave čeká na zahájení komunikace od typu Master, aby mohl přijímat a odesílat data přes linku SDA (Serial Data). Pro I^2C komunikaci se používá adresování, díky kterému ví, jaký komponent má odpovědět na komunikaci mezi typem Master a daným Slave [32].

Na prototypové desce bude dále napojen RTC (real-time clock) chronodot verze 2.0, Slave komponent na adrese 0x68, díky kterému bude modul moci určit reálný čas. Tento čas bude sloužit pro zápisy do paměti modulu jako záznam neoprávněné manipulace. Dalším komponentem se stane GY521, Slave komponent na adrese 0x69, sloužící jako akcelerometr a zároveň gyroskop. GY521 určuje, zda došlo ke zrychlení nebo vychýlení polohy chráněného zařízení. Implementují se ještě dva propojovací dráty (jumper wire), které budou napájeny na PIN D7 a D8, aby imitovaly situaci neoprávněného otevření skříňky. Pro obsluhu či uživatele zde budou realizovány LED (Light-Emitting Diode), dále jen světelné diody, na PIN D5 a D6, které budou sloužit jako ukazatel stavu chráněného zařízení. Na desce se bude napájet bezdrátová

komunikace NFC (Near Field Communication) Grove NFC v1.1, Slave komponent na adrese 0x24, který bude sloužit jako autentizační prvek pro ověření oprávněné osoby k zašifrovanému svazku. Autentizace bude probíhat pomocí unikátních identifikátorů nastavených v čipech, využívající standard Mifare, která převedou modul ze stavu hlídání do stavu klidového a naopak. Schéma lze vidět na obrázku 3.1, kde šedá barva značí adresy a jednotlivé PIN, modrá barva SCL a černá barva SDA.



Obr. 3.1: Schéma návrhu.

Samotná komunikace mezi zařízením a vytvořeným modulem se propojí pomocí sériové linky, kdy jedno zařízení posílá či vyměňuje data druhému zařízení bit po bitu, tedy sekvenčně, přes komunikační kanál.

3.3.2 Šifrování SD karty

Data, které bude sbírat zařízení Raspberry Pi 4 mohou být různorodá. Ať se jedná o videa nebo snímky z kamerového systému nebo soubory s obsahem citlivých dat, např. osobní údaje. Jelikož může dojít k šifrování velkých objemů dat, je zapotřebí, aby šifrovací algoritmus byl rychlý, dokázal zvládnout šifrovat velký objem dat s malým množstvím šíření chyb a aby k prolomení šifrovacího algoritmu došlo nejdříve za několik desítek let. Z těchto důvodu předešlé moduly používaly symetrickou blokovou šifru AES s 256bitovým klíčem a i v této práci se implementují do fyzického bezpečnostního modulu.

K šifrování oddílu a svazku na SD kartě se využije `Cryptsetup`, tedy utilita, pomocí které lze nastavit šifrování disku založené na modulu jádra DMCrypt. Cryptsetup podporuje svazek dm-crypt, LUKS (Linux Unified Key Setup), Loop-AES svazky poskytují omezenou podporu pro použití a jsou kompatibilní s TrueCrypt, VeraCrypt a BitLocker svazky [33].

Dm-crypt slouží pro vytváření, přístup, správu šifrovaných zařízení a device mapper, rámec poskytovaný linuxovým jádrem pro mapování fyzických blokových zařízení na virtuální bloková zařízeny, díky kterým dokáže šifrovat celé disky, a to i disky vyměnitelných médií, dále oddíly, svazky softwarového RAID (Redundant Array of Independent Disks), logické svazky i soubory. LUKS je používaný standard pro šifrování pevného disku Linux, tím usnadňuje kompatibilitu mezi distribucemi, ale také poskytuje bezpečnou správu více uživatelských hesel. LUKS ukládá všechny potřebné informace o nastavení v hlavičce oddílu, díky čemuž lze bezproblémově přenášet nebo migrovat data [34].

Cryptsetup nabízí dvě volby autentizace, tj. pomocí zadání hesla nebo obsahem souboru. V této práci bude autentizace probíhat pomocí obsahu souboru. Soubor bude AES s 256bitovým klíčem, který bude vygenerován z OpenSSL softwaru jako 64hexadecimálních znaků a následně uložen do konkrétního souboru, který se určí pro klíčový slot v cryptsetup pro šifrování a dešifrování svazku.

4 Implementace fyzického bezpečnostního modulu

Při implementaci vlastního fyzického bezpečnostního modulu se vychází z vlastního návrhu fyzického bezpečnostního modulu a z požadavků fyzického zabezpečení SD karty, který bylo stanoveno a popsáno v předchozí kapitole.

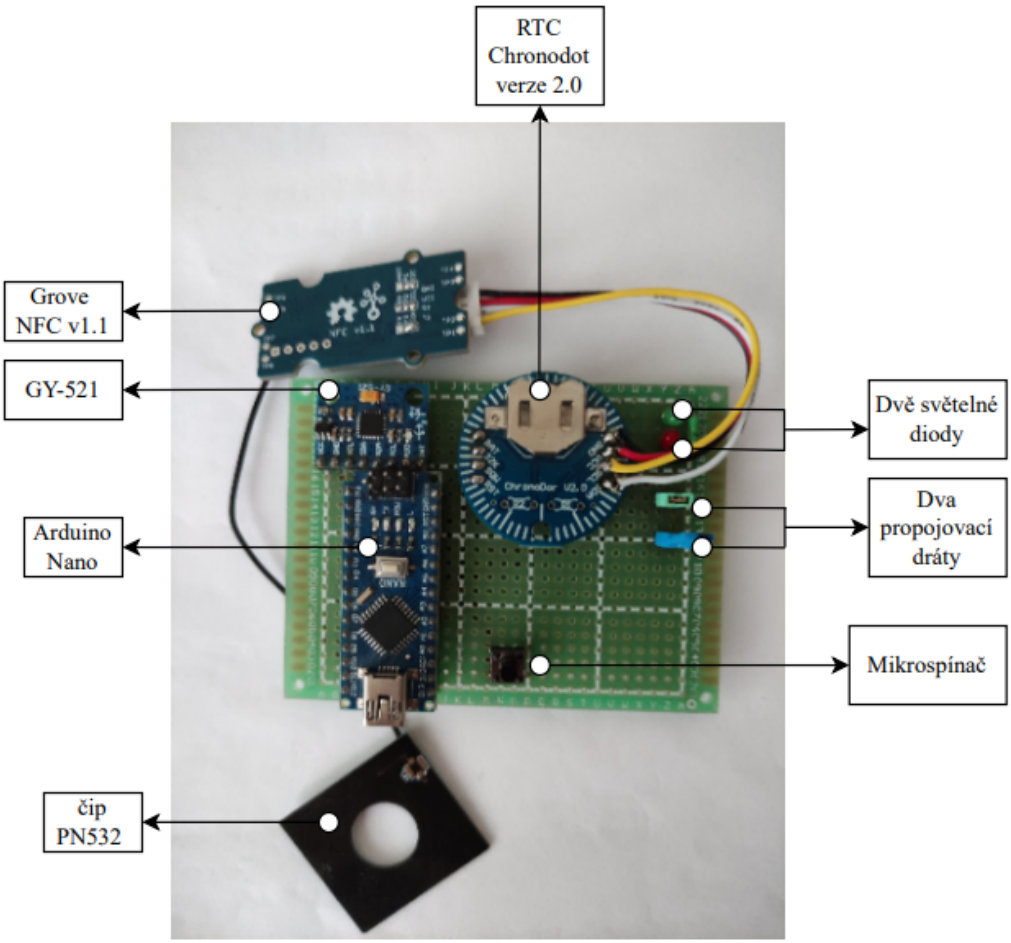
4.1 Sériová komunikace mezi Raspberry Pi a modulem

Při realizaci vlastního návrhu fyzického bezpečnostního modulu došlo k přidělání mikrospínače na prototypovou desku, který byl připojen na PIN D2. Díky tomu lze jednoduše přidávat karty. Ostatní komponenty, které byly popsány v podkapitole 3.3.1, zůstaly nepozměněné. Na obrázku 4.1 lze vidět vizuální implementaci s popisem a rozložením jednotlivých komponentů celého modulu na prototypové desce.

4.1.1 Šifrování oddílu při prvotní inicializaci a registrace čipových karet

Při sbírání dat bude fyzický bezpečnostní modul nastaven do režimu praktického nasazení, tedy nebude možné mít připojené periferie pro ovládání zařízení. Je zapotřebí, aby se v jazyce Python stal skript `raspberrySecure.py` automatizovaným, tedy při každém spuštění se zavede i samotný skript pro ochranu dat. Toho se docílí pomocí vytvoření souboru `secur.service` v adresáři `/lib/systemd/system`, který slouží jako úložiště konfigurace služeb systému `systemd` umožňující automatické zavedení služeb po startu systému. Nastavení souboru lze vidět na obrázku 4.2.

Ve skriptu `raspberrySecure.py` je vytvořená funkce `keyGenerate` pro vygenerování klíče, který slouží jako autentizační prvek pro šifrování oddílu. Pomocí příkazu `os.system(f"openssl enc -aes-256-cbc -pbkdf2 -k secret -P > {opensslFile}")` se vygeneruje 256bitový klíč, který se následně uloží do složky tmpfs souboru resp. `/tmp/finalKey.txt`. Tmpfs (Temporary File System) slouží jako dočasný úložný prostor v paměti, tedy jakmile se vypne zařízení, tak veškerá data budou ztracena. Tím pádem útočník, jenž odpojí Raspberry Pi 4 z napájení nebo odpojí SD kartu ze zařízení, nezíská klíč ani data, protože klíč se smaže z paměti a data bez klíče nebudou moci být dále dešifrována.



Obr. 4.1: Prototypový fyzický bezpečnostní modul

```
[Unit]
Description=secure app

[Service]
Type=simple
User=root
ExecStart=/usr/bin/python3 /home/pi/Desktop/raspberrySecure.py

[Install]
WantedBy=multi-user.target
```

Obr. 4.2: Nastavení souboru secur.service.

Při spuštění skriptu se nejprve zkontroluje, zda existuje cesta k oddílu a jestliže chybí, začne inicializační fáze skriptu, která vyvolá funkce `init`. Při této fázi se nej-

prve přidělí místo oddílu na disku a začne se šifrovat tento oddíl. Pomocí `luksFormat` se inicializuje oddíl LUKS a nastaví se počáteční přístupová fráze. Detailnější nastavení LUKS lze vidět na obrázku 4.5, kde nejzajímavější části je nastavený klíč v `Keyslots 0`. Poté se pomocí `luksOpen` otevře oddíl LUKS a nastaví se rámec pro správu svazků `secretMapper`. Po úspěšném ověření přístupové fráze, v tomto případě klíče vytvořeného pomocí nástroje openSSL softwaru, lze používat `secretMapper`. Toto lze vidět na obrázku 4.4. Dále se naformátuje `secretMapper`, díky kterému vznikne svazek sloužící pro zápis, čtení a vytváření souborů. Následně se vytvoří v `secretMapper` složka, kde se data budou sbírat. V posledním kroku se vykoná tzv. mount svazku a složky, aby byl dostupný operačnímu systému jako souborový systém pro čtení, zápis nebo obojí. Pomocí příkazu `lsblk` sloužícího k zobrazení podrobností o blokových zařízeních, lze vidět v `loop0` device mapper `secretMapper` a zašifrovaná složka `secretDirectory` viz obrázek 4.3. Kód pro prvotní inicializaci skriptu je ve výpisu 4.1.

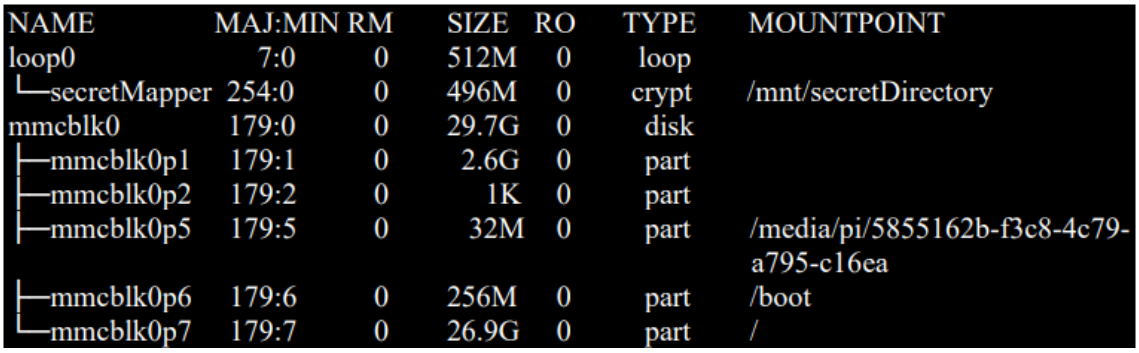
Ve skriptu `Arduino.ino` je vytvořená funkce `NFC`, která se vyvolá při prvotní inicializaci. Unikátní identifikátory čipových karet je možné registrovat do EEPROM buďto během 30 vteřin anebo uložením pěti různých čipových karet. Schéma algoritmu při prvotní inicializaci lze vidět na pravé straně obrázku 4.7.

Výpis 4.1: Prvotní inicializace LUKS.

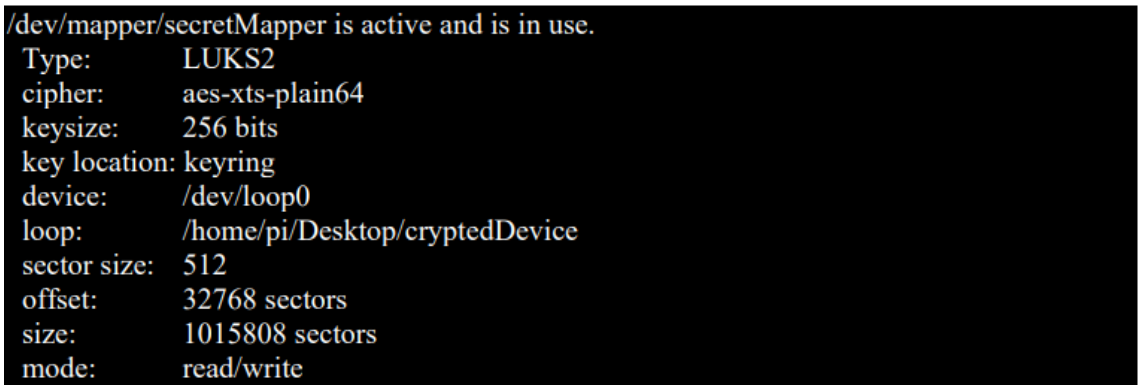
```
def init():
    os.system(f"fallocate -l 512M {device}")
    os.system(f"cryptsetup -v --type luks2 --cipher
    aes-xts-plain64 --key-size 256 --hash sha256
    --iter-time 2000 --use-urandom luksFormat {device}
    {keyFile} --keyfile-size 64")
    os.system(f"sudo cryptsetup luksOpen {device} secret
    --key-file {keyFile} --keyfile-size 64")
    os.system(f"sudo mkfs.ext4 -j {container}")
    os.system(f"sudo mkdir {directory}")
    os.system(f"sudo mount {container} {directory}")
```

4.1.2 Odeslání vygenerovaného klíče do modulu

Výměna klíče z Raspberry Pi 4 do modulu je velmi důležitou fází, při které nesmí dojít k žádné chybě při přenosu, jelikož následně bude tento klíč sloužit po celou dobu komunikace jako autentizační prvek pro svazek. Klíč se přeposílá ze zařízení pouze při prvotní inicializaci.



Obr. 4.3: Zašifrovaný svazek.

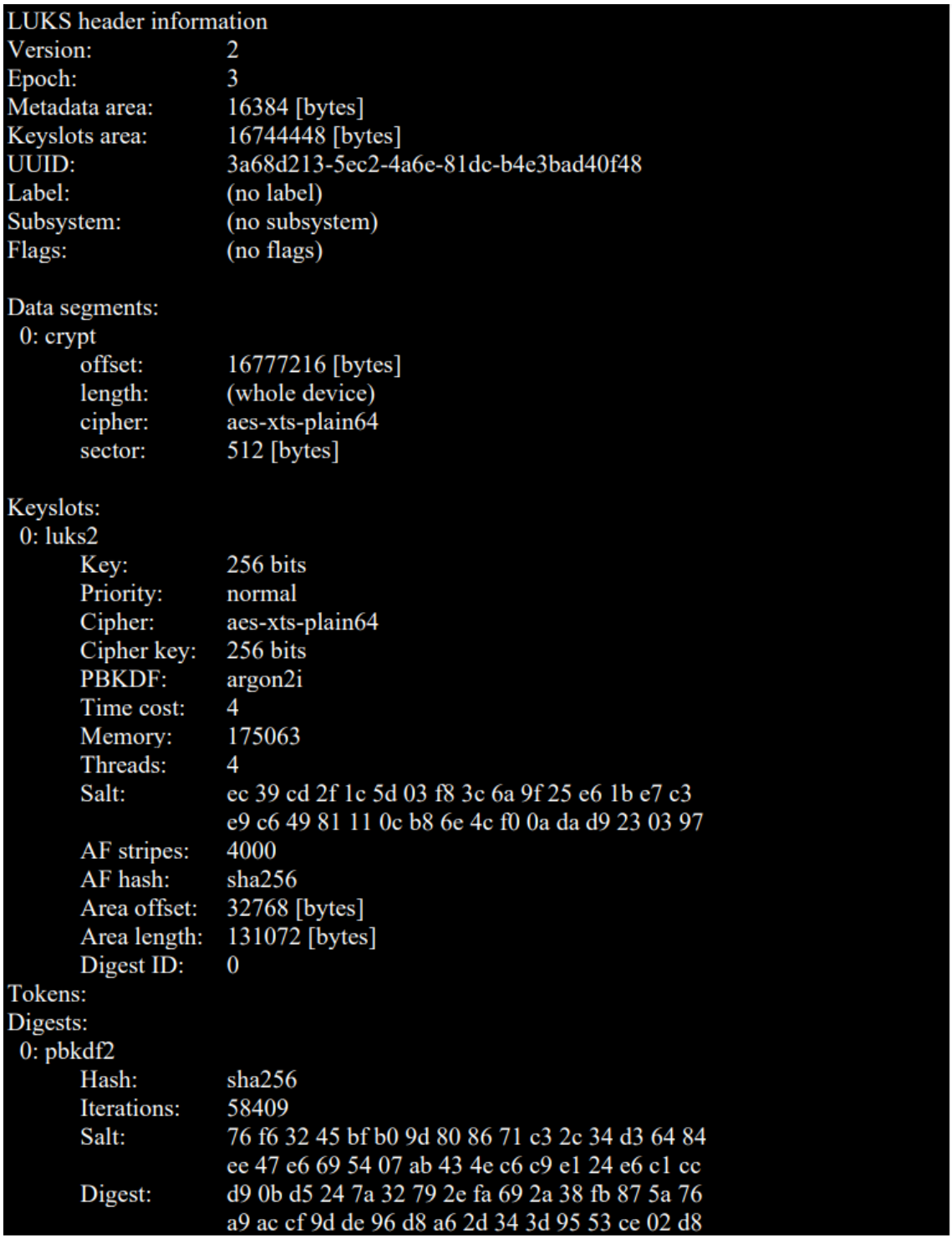


Obr. 4.4: Podrobnější stav mapování nástroje LUKS.

Komunikace mezi Raspberry Pi 4 a vytvořeným modulem probíhá na sériové lince, viz obrázek 4.6. Klíč, který se vygeneroval v zařízení je přes sériovou linku přeposlán do modulu. Ve skriptu `Arduino.ino`, který je určen pro modul, lze nalézt funkci `readSerialPort`, která naslouchá a čeká na příchozí komunikaci od zařízení Raspberry Pi 4. Jestliže klíč, vygenerovaný zařízením, se přepošle přes sériovou linku, tak si modul tento klíč převezme a uloží pomocí funkce `setKey` do EEPROM (Electrically Erasable Programmable Read-Only Memory). EEPROM je stabilní a energeticky nezávislé paměťové zařízení, které se používá pro ukládání minimálního množství dat v počítačových a elektronických systémech a zařízeních, jako jsou desky plošných spojů a nevyžaduje žádný typ pohotovostního napájení.

4.1.3 Šifrování svazku při dalším spuštění skriptu

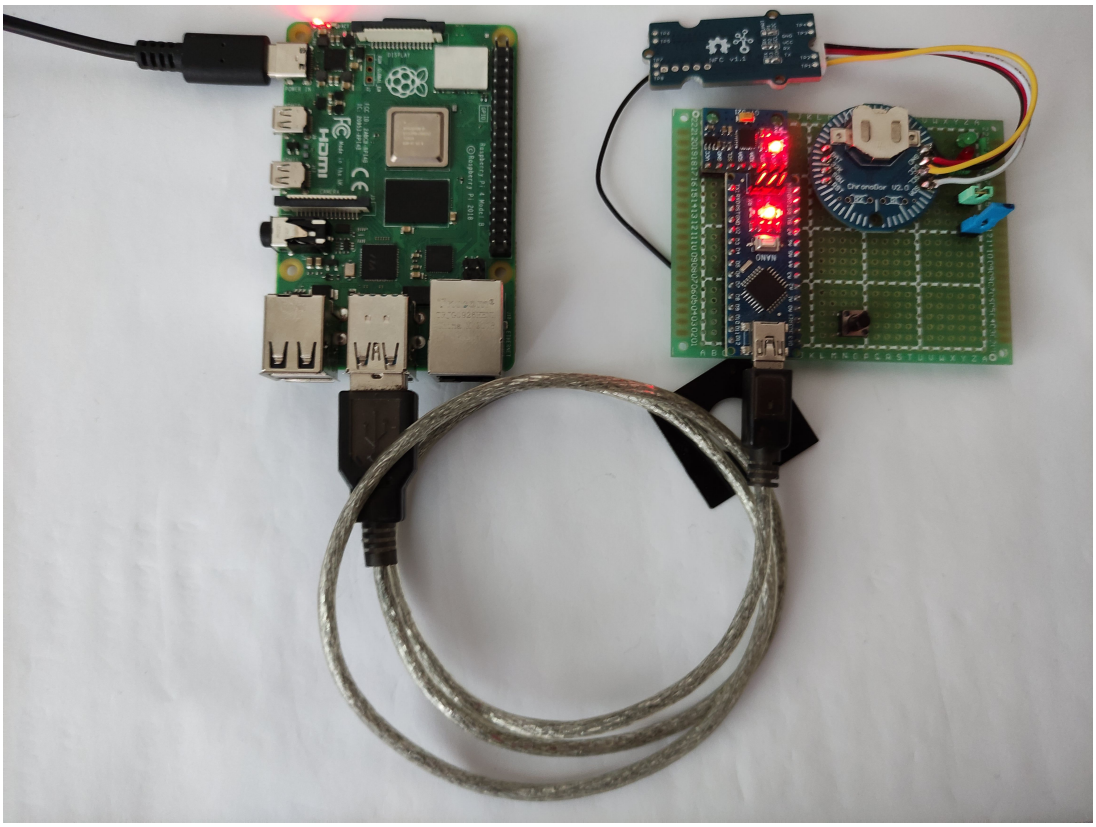
Kdykoli dojde k výpadku proudu nebo autorizovaná osoba vyjme SD kartu a vloží ji zpět, dojde ke ztrátě klíče v zařízení, protože je uložen v dočasné úložné paměti. Odeslaný vygenerovaný klíč z Raspberry Pi 4 do modulu je možné získat zpět pomocí



Obr. 4.5: Výpis informací v záhlaví zařízení LUKS.

funkce v modulu `getKey`.

Komunikace funguje na bázi zpráv znaků 0 až 2, které jsou poslány ze zařízení při zapnutí skriptu. Znak 0 značí inicializační fázi z podkapitole 4.1.1, modul má



Obr. 4.6: Sériová komunikace mezi Raspberry Pi 4 a modulem.

naslouchat komunikaci, uložit si klíč do EEPROM a rozsvítit zelenou světelnou diodu. Znak 1 značí, že je vše v pořádku, cesta v zařízení ke klíči a k svazku existuje, a tím pádem není třeba něco posílat modulu nebo od modulu něco dostat a modul svítí zeleně. Znak 2 značí případ, kdy došlo ke ztrátě klíče, tedy buď k výpadku zařízení, nebo autorizovaná osoba vytáhla SD kartu a dala ji zpět. V tento moment neexistuje cesta ke klíči a je zapotřebí, aby byl poslán klíč z modulu do zařízení, jinak by nebylo již možné nadále sbírat data do zašifrovaného svazku. Po přijetí klíče si Raspberry Pi 4 klíč uloží opět do své paměti `tmp/finalKey.txt` a vykoná mount na složku pomocí funkce `openDirectory`, aby mohl opět fungovat a sbírat data do zašifrované složky. Celé schéma algoritmu lze vidět na obrázku 4.7, kde začíná start na straně Raspberry Pi 4, které si žádá informace o situaci okolí a Arduino Nano tyto informace dále posílá do modulu. Tyto případy popsané výše jsou ukázány na schématu před začátkem cyklu.

4.1.4 Neoprávněná manipulace se zařízením

Jakmile dojde k jakémukoliv narušení, tedy s Raspberry Pi 4 dojde k jakékoliv manipulaci, tj. otevření skříňky nebo pohybu skříňky či je špatně použit klíč, tak se

provede tzv. umount složky a svazek LUKS se zavře. Tyto kroky lze vidět na schématu 4.7 na začátku cyklu až po konec programu.

Pro rozpoznání, zda došlo k neoprávněné manipulaci ze zařízením, slouží v modulu akcelerometr, gyroskop a dva propojovací dráty. Ve skriptu `Arduino.ino` funkce `setup`, který se vykoná vždy při zapnutí skriptu, lze nalézt prvotní nastavení os v akcelerometru a gyroskopu. Je to z toho důvodu, aby si osoba, která implementuje tento modul do jiné krabice a má odlišné rozměry krabice, mohla upravit a nastavit polohu modulu podle vlastních potřeb před zapnutí skriptu. Následně ve funkci `loop`, který je nekonečným cyklem, se vyvolává funkce `accelerometer`, kde se sbírají stejné údaje, jak v `setup` s rozdílem, že tato funkce je vyvolaná v každém cyklu. Cyklus má za úkol neustále srovnávat hodnoty v `setup`, jenž se nastavily při zapnutí skriptu. V případě zaznamenaného pohybu, ke kterému by mohlo dojít nedopatřením, je vytvořeno počítadlo, které je nastavené na nule a časovač, který se spustí, jakmile dojde k pohybu. Časovač počítá do šesti sekund, pokud se do šesti sekund vyhodnotí pohyb jako nekonzistentní, dojde k navrácení hodnoty na nulu. Jestliže pohyb přetrvává a dochází k většímu odchýlení hodnot akcelerometru nebo gyroskopu, vyhodnotí se tato situace jako pokus vloupání do skřínky. Modul zašle tuto informaci do Raspberry Pi 4, aby vykonal umount složky a následně svazku. V zařízení se vytvoří záznam, který zapíše tuto informaci a následně se pošle e-mailová notifikace obsluze nebo uživateli. Následně Raspberry Pi 4 pošle znak 0, který značí, že se někdo pokusil dostat do krabice. Modul převezme znak 0, a ten si uloží do EEPROM s časem, pomocí RTC komponentu, kdy tomu došlo. Díky znaku 0 modul ví, že si má pomocí funkce `randomHexa` do EEPROM zapsat náhodné hexadecimální bajtové znaky a nahradit tím klíč, který sloužil jako šifrování a dešifrování svazku. Klíč bude nepoužitelný a nadále se již nebude moc využívat pro otevření svazku.

Na prototypové desce jsou dále napájeny dva propojovací dráty, sloužící pro kontrolu skřínky, zdali nedošlo k jejímu otevření. Jestliže se jeden z propojovacích drátu rozpojí, modul pošle tuto informaci do Raspberry Pi 4 a opět dojde k vykonání umount složky a svazku, informace o proniknutí do krabice se zapíše do souboru a pošle se e-mailová notifikace obsluze či uživateli. Raspberry Pi 4 do modulu zpátky pošle znak 1, který značí otevření skříňky. Tento znak se s časem zapíše do záznamu v modulu. To samé platí i se znakem 1, kdy dojde k přepsání hodnot správného klíče na náhodné hexadecimální bajty v EEPROM, a tím již nebude možné otevřít šifrovaný svazek s daty.

Posledním znakem je 2, který slouží pro kontrolu klíče. Na to je v modulu vytvořená funkce `bool compareKey` a v `loop` vytvořený časovač, který každé dvě minuty neustále kontroluje správnost klíče. Modul žádá v určitých intervalech zařízení pro uložený klíč v tmpfs souboru. Pokud klíč není shodný s klíčem uložený v modulu resp. EEPROM, dojde k tomu, že se pošle znak 2 do modulu. Ten značí nepravost

klíče pro šifrování oddílu. V Raspberry Pi 4 se zapíše záznam o nepravosti klíče, vykoná se umount složky a svazku, zapíše se do souboru informace o nepravosti klíče a opět se pošle e-mailová notifikace obsluze či uživateli. Znak 2 se pošle do modulu, který přepíše opět hodnotu klíče na náhodné hexadecimální bajty a zapíše záznam s časem do EEPROM.

4.1.5 Oznámení obsluze o neoprávněné manipulaci se zařízením

Oznámení obsluze či uživateli o vniku do skříňky probíhá dvěma způsoby – vizuálně u zařízení, nebo notifikacemi přes e-mailovou zprávu. Princip vizuálního zobrazení u zařízení funguje pomocí dvou světelných diod, červené a zelené, implementované na modulu sloužící pro zobrazení určitých stavů, ve kterém se právě nachází Raspberry Pi 4. Jestliže svítí pouze zelená světelná dioda, značí to, že je ve stavu mount svazku, a tím pádem sbírá data do zašifrované složky, resp. modul funguje správně. Pokud se rozsvítí červená světelná dioda, znamená to, že zařízení je ve stavu umount svazku, tím pádem nesbírá data do zašifrované složky, resp. modul nepracuje správně a je zapotřebí, aby obsluha jednala.

Jestliže svítí obě světelné diody zároveň, znamená to, že se zavádí operační systém do zařízení a připravují se potřebné kroky k otevření svazku. Při prvotní inicializaci systému to trvá déle, jelikož se zde vytváří místo na disku a formátuje se oddíl na svazek se složkou. V moment, kdy svítí obě světelné diody obsluha čeká, dokud nezhasne červená dioda. V momentě, kdy blikají obě světelné diody v krátkém časovém úseku, znamená to, že klíč není v Raspberrry Pi 4 totožný s klíčem v modulu - tedy mohlo dojít k problému při přenosu, nebo klíč byl nějakým způsobem poškozen. Tento případ je velmi ojedinělý a je velmi malá pravděpodobnost, že by se tato situace uskutečnila.

Pokud svítí pouze červená světelná dioda, dochází k tomu, že se zařízením se někdo pokusil manipulovat. Buď se snažil útočník otevřít skřínku, což by zaznamenal akcelerometr, jelikož by došlo k většímu výkyvu hodnot, nebo by došlo k jejímu otevření pomocí propojovacích drátů, které by byly rozpojeny. Při rozsvícení pouze zelené světelné diody nemusí obsluha nic vykonávat, jelikož je vše v pořádku.

V případě, že by se jednalo o místech, kde by bylo složité kontrolovat stav zařízení vizuálně, je vytvořená funkce `email_sender`, která slouží jako e-mailová notifikace. Pověřená osoba dostane e-mailovou zprávu s časem a událostí, která nastala při neoprávněné manipulaci.

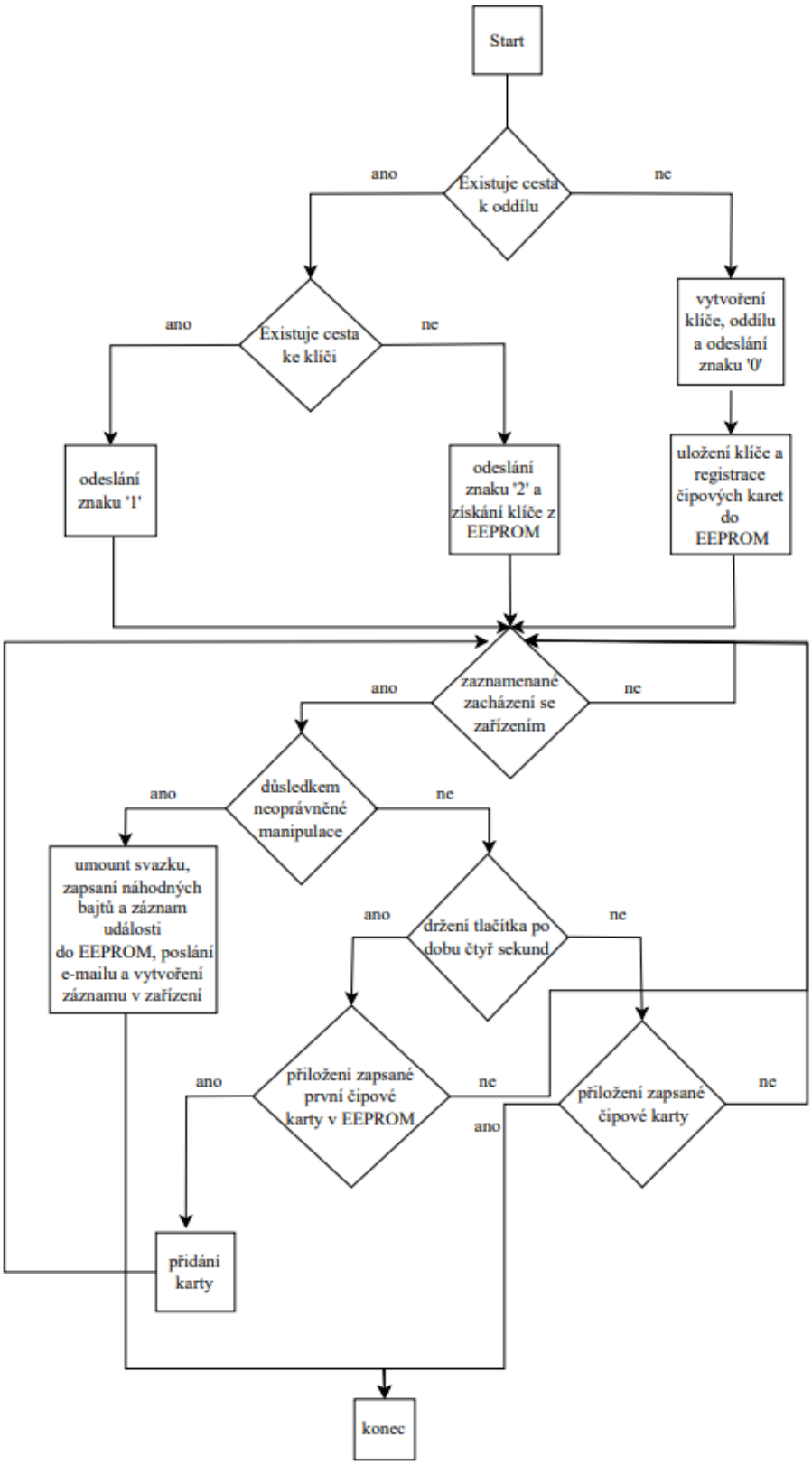
4.1.6 Autorizace a oznámení pro oprávněného uživatele

K autorizaci oprávněného uživatele dochází pomocí unikátních čísel čipových karet uložených v EEPROM – umožňuje autorizované osobě komunikovat s modulem.

Díky čipům přejde modul z hlídacího stavu do klidového stavu, tedy celý modul se vypne a na zašifrovaném svazku se vykoná umount. Se zařízením je následně možné manipulovat jakkoliv bez ztráty klíče. Klidový stav rozezná uživatel tak, že jsou obě světelné diody zhasnuté. Hlídací stav naopak uživatel pozná tak, pokud opět bude svítit zelená světelná dioda.

Během inicializační fáze blikají obě světelné diody zároveň, a to v delším časovém úseku. V momentě, kdy bude přiložena karta a zasvítí světelné diody, znamená to, že unikátní identifikátor čipu se uložil do EEPROM a lze přiložit další kartu pro registraci. Po inicializační fázi je již možné se autentizovat pomocí čipových karet. Pokud dvě sekundy svítí červená světelná dioda, dojde k tomu, že čip není správný. V zařízení je soubor se záznamem, kde se ukládá čas a neoprávněná karta s identifikačním číslem a ten se také pošle jako e-mailová notifikace uživateli.

Čipové karty je možné také přidávat v průběhu běžného stavu. První čip, který je zapsán do EEPROM je hlavní čip, který umožňuje zapisování karet. Proto je důležité dbát na první kartu, která je zapsána. Při zapsání karty je nutné držet po dobu čtyř sekund mikrospínač implementovaný na modulu. Tím se zapne časovač pro přidávání karet. První tři sekundy je nutné přiložit první čip uložený v EEPROM, tedy čip, který byl přiložen jako první. Pokud bude čip totožný. bude svítit zelená světelná dioda. Následně je možné po dobu deseti sekund přidávat karty. Přidanou kartu lze poznat rychlým přebliknutím na červenou světelnou diodu a zpátky na zelenou světelnou diodu. Proces tohoto stavu lze vidět na schématu 4.7.



Obr. 4.7: Schéma algoritmu možných stavů zařízení.

5 Výhody a nevýhody fyzického bezpečnostního modulu

Vlastní fyzický bezpečnostní modul byl inspirovaný již vytvořenými moduly, tj. Zymkey 4i a SD kartou Swissbit, které byly popsaný v kapitole 2 a došlo k eliminaci nedostatků, popsány v podkapitoly 3.1.1. Tato kapitola pojednává o výhodách a nevýhodách implementace vlastního fyzického bezpečnostního modulu a také jaké nevýhody vznikly a nastaly během implementace vlastního modulu. Závěrem kapitoly jsou rozebrány nevýhody návrhu a popsány, jak by mohly být odstraněny.

5.1 Výhody fyzického bezpečnostního modulu

Nejprve bylo za úkol vlastní fyzický bezpečnostní modul vytvořit tak, aby bylo možné modul implementovat na různá zařízení s různými datovými úložišti, aniž by došlo ke ztrátě šifrovaných dat či k jejímu odcizení. Toho se dosáhlo pomocí modulu, který nemá žádnou fyzickou vazbu na SD kartu a ani žádnou fyzickou vazbu na zařízení Raspberry Pi 4. Jelikož modul nemá žádnou fyzickou vazba na SD kartu, tak automaticky je umožněno uživateli iniciovat libovolné změny datového úložiště v zařízení, a tedy i škálovat různé objemy paměťových karet. Kdyby došlo k poškození zařízení, nikoliv poškození SD karty, tak ho lze také jednoduše vyměnit a není třeba vytvářet nový modul. To vše bylo umožněné pomocí utility cryptsetup, který šifruje data na disku.

Další výhodou vlastního modulu najdeme v desce Arduino Nano, který má nízkou úroveň spotřeby. Pokud by došlo k výpadku proudu, tak by došlo k vypnutí Raspberry Pi 4 a ke ztrátě šifrovaných dat. Ke ztrátě šifrovaných dat ale nedojde, jelikož klíč, který šifruje a dešifruje svazek, je uložen v desce Arduino Nano, resp. EEPROM.

Dalším bonusem jsou viditelné stavy zařízení. V Zymkey 4i sloužila pouze jedna modrá světelná dioda, která byla implementována na modulu. SD karta Swissbit neměla žádné vizuální upozornění, v jakém stavu se právě nachází. Ve vlastním modulu tuto skuteřnost umožňují dvě světelné diody, které jsou popsány do detailu v podkapitole 4.1.5.

V poslední řadě je vyřešen problém při odcizení SD karty. Při odcizení SD karty, kde není možný přístup k internetu, nastane situace taková, že uživatel nemá ponětí, co a kdy se stalo se zařízením. Tento problém je vyřešen pomocí paměti v modulu, který uchovává stavy zařízení. Jestliže dojde k nějaké neoprávněné manipulaci, stane se to, že se uloží do EEPROM stav s časem. Díky tomu lze jednoduše určit, kdy došlo k činu a odkdy se data neuchovávají v zařízení.

5.2 Nevýhody fyzického bezpečnostního modulu

Největší nevýhodou a zároveň výhodou, popsáno v podkapitoly 5.1, vlastního fyzického bezpečnostního modulu, je právě ukládání klíče do EEPROM. EEPROM neumožňuje žádnou autorizaci a ani autentizaci uživatele, a právě proto se stává také nejzranitelnější části celé práce. Když se útočník dostane do krabice a vytáhne s SD kartu i zabezpečovací modul, docílí klíče, který je uchován v EEPROM, sloužící k šifrování svazku. Problém je vyřešen pomocí funkce `randomHexa`, která zaplní EEPROM náhodnými hexadecimálními bajty. Náhodné bajty přepíší uložený klíč a také zaplní EEPROM tak, aby se nedal vyčíst správný klíč. Nevýhoda spočívá v její omezené životnosti. Je vytvořená tak, aby umožnila 100 000 cyklů čtení a mazání [35]. Může tedy dojít k jejímu brzkému opotřebení při velkém přepisování, čtení či mazání dat. Problém by mohl být vyřešený pomocí mikroprocesorového čipu, který je podle návrhu chráněn před neoprávněným přístupem a používá se ke spouštění omezené sady aplikací a také k ukládání důvěrných a kryptografických dat jako je například UICC (Univerzální karta s integrovaným obvodem).

Druhou nevýhodou je používání čipových karet jako autentizační prvek. Čipové karty nevyužívají žádný autentizační prvek, tedy když budou ztraceny nebo odcizeny, můžou být zneužity. Tento problém by se dal vyřešit pomocí ověření autentizace uživatele pomocí mobilního telefonu, ať již je to gesto nebo PIN, kde by byla vytvořená mobilní aplikace, která by dokázala komunikovat s NFC komponentem.

Další možnou věcí, kterou by se dalo vylepšit modul, je naimplementování kamery určenou pro Raspberry Pi nebo pro vytvořený modul. Kamera by vytvářela snímky, jakmile by došlo k otevření krabice a ukládala tyto snímky na SD kartu nebo posílala jako e-mailová notifikace.

Závěr

Hlavním cílem této bakalářské práce bylo navrhnout a realizovat fyzický bezpečnostní modul, který chrání citlivá data uložená na platformě Raspberry Pi v otevřeném prostranství, kde hrozí vyjmutí SD karty a následně zneužití dat, která jsou uložená v úložišti. Při tvorbě fyzického bezpečnostního modulu se primárně zaměřovalo na implementaci senzorů a algoritmů, které by rozpoznaly neoprávněnou manipulaci.

V rámci teoretické části se práce zabývá zabezpečením senzorů v IoT. Byly představeny funkce IoT zařízení a rozebrány některé možné bezpečnostní zranitelnosti z různých odvětví.

Nejprve byly v praktické části teoreticky zkoumány a prakticky předvedeny dva doposud existující zabezpečovací moduly – Zymkey 4i od společnosti Zymbit a micro SD karta Swissbit PS-45u od společnosti Swissbit.

Zymkey 4i nabízí množství pokročilých funkcí. Pro šifrování dat v Raspberry Pi se využívá šifrovacího algoritmus AES s 256bitovým klíčem. Proti fyzickému vniknutí je zde možnost využít perimetr, a to k zjištění možnosti poškození obvodu z přívodu a akcelerometr sloužící k upozornění na pohyb či dotyk daného zařízení. Autentizace Zymkey 4i probíhá pouze v případě, že je nastavena v produkčním režimu a to tak, že dostane jedinečnou a nezměnitelnou identitu, která je užívána k ověřování následných interakcí se zařízením.

Micro SD karta Swissbit PS-45u využívá také šifrovací algoritmus AES s 256bitovým klíčem. Autentizace probíhá během startovací fáze před spuštěním systému Swissbit třemi způsoby – PIN, USB, NET Policy Server. NET Policy Server je nejefektivnější autentizační ochrana, protože využívá zabezpečení lokálního úložiště před neoprávněnou manipulací a uzamknutí zařízení na vzdálenost.

Cílem práce bylo vytvořit nový prototypový fyzický bezpečnostní modul, který umí stejně a efektivně chránit citlivá data uložená na SD kartě, jako to dokázaly předešlé moduly, a to před neoprávněnou osobou a zároveň eliminovat nevýhody, které byly u těchto dvou zařízení.

Vytvořený vlastní fyzický bezpečnostní modul dokáže detekovat otevření skříňky, neoprávněnou manipulaci se skříňkou, ukládat šifrovací klíč do modulu a ověřovat autentizaci pomocí čipových karet, které podporují NFC komunikaci. V jakém stavu se právě nachází modul lze jednoduše poznat pomocí dvou světelných diod. V momentě, kdy dojde k jakémukoliv narušení zařízení, se uloží do EEPROM příslušný čas a stav. V případě internetového připojení dostane také uživatel či obsluha e-mailovou notifikaci.

Vlastní fyzický bezpečnostní modul byl implementován na protypovou desku. Modul by mohl být dále doplněn například kamerou, která by vytvářela snímky,

jakmile by došlo k otevření krabice. Další věcí, kterou by bylo možné vylepšit, je výměna čipových karet za mobilní telefonní aplikaci, protože mobilní telefon skýtá mnohem více prostředků, které dokáží ověřit totožnost uživatele, a tím zdokonalit bezpečnostní systém. Nakonec by bylo také možno externě připojit SAM modul, který pro uložení klíče používá kryptograficky zabezpečené úložiště.

Literatura

[1] SUJAY VAILSHERY, Lionel. *Internet of Things (IoT) and non-IoT active device connections worldwide from 2010 to 2025. Https://www.statista.com* [online]. 2021 [cit. 2021-11-15]. Dostupné z: <https://www.statista.com/statistics/1101442/iot-number-of-connected-devices-worldwide/>.

[2] *Zymbit_ProductBrief_Zymkey4i_04100100_04150911_B1. Https://www.zymbit.com/* [online]. 2021 [cit. 2021-11-15]. Dostupné z: <https://www.zymbit.com/wp-content/uploads/2021/08/Zymbit_ProductBrief_Zymkey4i_04100100_04150911_B1.pdf>

[3] *Secure elements in use by Swissbit. Https://www.swissbit.com* [online]. [cit. 2021-11-15]. Dostupné z: <https://www.swissbit.com/en/products/security-products/security-editions/#secure-microsd>.

[4] THOMAS, Mike. *27 Top Internet of Things Examples You Should Know. Https://builtin.com/* [online]. 2021 [cit. 2021-11-29]. Dostupné z: <https://builtin.com/internet-things/iot-examples>.

[5] ERRABELLY, Ranadheer, Kewei SHA, Wei WEI a T. Andrew YANG. *EdgeSec: Design of an Edge Layer Security Service to Enhance IoT Security. Https://ieeexplore.ieee.org* [online]. 2017 [cit. 2021-11-15]. Dostupné z: <https://ieeexplore.ieee.org/document/8014363>.

[6] DICKSON, Ben. *Why you need to worry about your smart-home's security? Https://www.iotsecurityfoundation.org/ /* [online]. 2016 [cit. 2021-11-15]. Dostupné z: <https://www.iotsecurityfoundation.org/why-you-need-to-worry-about-your-smart-homes-security/>.

[7] *Why IoT Security Is Important for Your Home Network. Https://www.kaspersky.com/* [online].[cit. 2021-11-15]. Dostupné z: <https://www.kaspersky.com/resource-center/threats/secure-iot-devices-on-your-home-network>

[8] *5 Ways Smart City Technology Benefits Cities and Residents. Https://skyfii.io/* [online]. [cit. 2021-11-15]. Dostupné z: <https://skyfii.io/blog/5-ways-smart-city-technology-benefits-cities-and-residents/>

[9] *Smart Cities: Threat and Countermeasures. Https://www.rambus.com* [online]. [cit. 2021-11-15]. Dostupné z: <https://www.rambus.com/iot/smart-cities/>

[10] *10 INTERNET OF THINGS (IOT) HEALTHCARE EXAMPLES, AND WHY THEIR SECURITY MATTERS. Https://ordr.net/* [online]. [cit. 2021-11-15]. Dostupné z: <https://ordr.net/article/iot-healthcare-examples/>

[11] *How Does PKI Work? Https://www.venafi.com* [online]. [cit. 2021-12-11]. Dostupné z: <https://www.venafi.com/education-center/pki/how-does-pki-work>

[12] *Top 10 Vulnerabilities that Make IoT Devices Insecure. Https://www.venafi.com* [online]. [cit. 2021-12-11]. Dostupné z: <https://www.venafi.com/blog/top-10-vulnerabilities-make-iot-devices-insecure>

[13] *Key PKI benefits. Https://www.anz.com* [online]. [cit. 2021-12-11]. Dostupné z: <https://www.anz.com/institutional/online-security/public-key-infrastructure/key-pki-benefits/>

[14] SHEA, Sharon a Ivy WIGMORE, ed. *Top 10 Vulnerabilities that Make IoT Devices Insecure. Https://internetofthingsagenda.techtarget.com* [online]. 2021 [cit. 2021-12-11]. Dostupné z: <https://internetofthingsagenda.techtarget.com/definition/IoT-security-Internet-of-Things-security>

[15] WALKOWSKI, Debbie a Shahnawaz BACKER. *Securing APIs: 10 Ways to Keep Your Data and Infrastructure Safe. Https://www.f5.com* [online]. 2020 [cit. 2021-12-11]. Dostupné z: <https://www.f5.com/labs/articles/education/securing-apis--10-best-practices-for-keeping-your-data-and-infra

[16] *What is a Raspberry Pi? Https://opensource.com*[online]. [cit. 2021-12-11]. Dostupné z: <https://opensource.com/resources/raspberry-pi>

[17] *What is a Raspberry Pi? Https://www.raspberrypi.org*[online]. [cit. 2021-12-11]. Dostupné z: <https://www.raspberrypi.org/help/what-%20is-a-raspberry-pi/>

[18] HATTERSLEY, Lucy. *Raspberry Pi 4 vs Raspberry Pi 3B+. Https://magpi.raspberrypi.com* [online]. 2019 [cit. 2021-12-11]. Dostupné z: <https://magpi.raspberrypi.com/articles/raspberry-pi-4-vs-raspberry-pi-3b-plus>

[19] BORGINI, Julia. *Don't forget IoT physical security when planning protection. Https://internetofthingsagenda.techtarget.com/* [online]. 2020 [cit. 2021-11-15]. Dostupné z: <https://internetofthingsagenda.techtarget.com/tip/Dont-forget-IoT-physical-security-when-planning-protection>

[20] *Secure your digital assets from cyber physical exploits.* [online]. [cit. 2021-11-15]. Dostupné z: <https://www.zymbit.com/>

[21] *Security Modules. Https://www.zymbit.com/* [online]. [cit. 2021-11-15]. Dostupné z: <https://www.zymbit.com/security-modules/>

[22] *HSM4 a HSM6. Https://www.zymbit.com* [online]. [cit. 2021-12-08]. Dostupné z: <https://www.zymbit.com/wp-content/uploads/2021/09/HSM4-Hero-Cutaway-BW-1000x750-NEW_banner.jpg>

[23] *Python API Documentation. Https://docs.zymbit.com/* [online]. [cit. 2021-11-18]. Dostupné z: <https://docs.zymbit.com/api/python_api/>

[24] *Enabling Production Mode. Https://docs.zymbit.com/* [online]. [cit. 2021-11-18]. Dostupné z: <https://docs.zymbit.com/getting-started/zymkey4/production-mode/>

[25] *Company Profile. Https://www.swissbit.com* [online]. [cit. 2021-12-06]. Dostupné z: <https://www.swissbit.com/en/company/about-us/>

[26] *Binding, Device ID, and Authentication. Https://docs.zymbit.com* [online]. [cit. 2021-12-08]. Dostupné z: <https://docs.zymbit.com/reference/binding/>

[27] *Why Swissbit. Https://www.swissbit.com* [online]. [cit. 2021-12-06]. Dostupné z: <https://www.swissbit.com/en/company/why-swissbit/>

[28] *Solutions for the Internet of Things. Https://www.swissbit.com* [online]. [cit. 2021-12-06]. Dostupné z: <https://www.swissbit.com/en/solutions/iot/>

[29] *Secure Boot Solution for Raspberry Pi. Https://www.swissbit.com*[online]. [cit. 2021-12-06]. Dostupné z: <https://www.swissbit.com/en/products/security-products/secure-boot-solution/>

[30] *pSFSD032GN3PM1TO-I-HG-020-RP0. Https://ch.farnell.com* [online]. [cit. 2021-12-13]. Dostupné z: <https://ch.farnell.com/swissbit/sfsd032gn3pm1to-i-hg-020-rp0/32gb-microsd-karte-raspberry-pi/dp/3489485?CMP=GRHB-OCTOPART>

[31] *ZYMKEY 4I. Https://ch.farnell.com* [online]. [cit. 2021-12-13]. Dostupné z: <https://ch.farnell.com/zymbit/zymkey-4i/iot-security-modul-i2c-raspberry/dp/2947716>

[32] CAMPBELL, Scott. *BASICS OF THE I2C COMMUNICATION PROTOCOL. Https://www.circuitbasics.com/* [online]. [cit. 2022-04-13]. Dostupné z: <https://www.circuitbasics.com/basics-of-the-i2c-communication-protocol/>

[33] *Cryptsetup(8) — Linux manual page. Https://man7.org* [online]. [cit. 2022-04-13]. Dostupné z: <https://man7.org/linux/man-pages/man8/cryptsetup.8.html>

[34] *Cryptsetup. Https://gitlab.com* [online]. [cit. 2022-04-13]. Dostupné z: <https://gitlab.com/cryptsetup/cryptsetup>

[35] COBURN, JOE. *How to Use Arduino EEPROM to Save Data Between Power Cycles. Https://www.makeuseof.com* [online]. 13.12.2016 [cit. 2022-05-10]. Dostupné z: <https://www.makeuseof.com/tag/use-arduino-eeprom-save-data-power-cycles/>

Seznam symbolů a zkratek

IoT	Internet věcí
ECDSA	Protokol digitálního podpisu s využitím eliptických křivek
ECDH	Diffieho–Hellmanův protokol s využitím eliptických křivek
AES	Advanced Encryption Standard
SHA	Secure Hash Algorithm
SAM	Secure Access Module
SSH	Secure Shell
WPA	chráněný přístup k Wi-Fi
TELNET	Teletype Network
FTP	File Transfer Protocol
HTTP	Hypertext Transfer Protocol
API	Application Programming Interface
PKI	Public Key Infrastructure
GPIO	Univerzální vstupní/výstupní pin
TRNG	Hardwarový generátor náhodných čísel
LUKS	Linux Unified Key Setup
BLOB	Binary Large Object
ID	Identifikace
NVRAM	Non-Volatile Random-Access Memory
PIN	Identifikační číslo
SDK	Software development kit
USB	Univerzální sériová sběrnice
GUI	Grafické uživatelské rozhraní
IP	Internet Protocol

SD	Secure Digital
SCL	Serial Clock Line
SDA	Serial Data
I^{2}C	Inter-Integrated Circuit
RTC	real-time clock
LED	Elektroluminiscenční dioda
NFC	Near Field Communication
RAID	Redundant Array of Independent Disks
EEPROM	Electrically Erasable Programmable Read-Only Memory
UICC	Univerzální karta s integrovaným obvodem