VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky
a komunikačních technologií

DIPLOMOVÁ PRÁCE

Brno, 2018 Bc. Jiří Kubů

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

ELEKTRONICKÝ ZABEZPEČOVACÍ SYSTÉM S PRVKY IOT
SECURITY ALARM WITH IOT FEATURES

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR
Bc. Jiří Kubů

VEDOUCÍ PRÁCE
SUPERVISOR
Ing. Ondřej Krajsa, Ph.D.

BRNO 2018

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Diplomová práce

magisterský navazující studijní obor **Telekomunikační a informační technika**
Ústav telekomunikací

Student: Bc. Jiří Kubů ***ID:*** 154783
Ročník: 2 ***Akademický rok:*** 2017/18

NÁZEV TÉMATU:

Elektronický zabezpečovací systém s prvky IoT

POKYNY PRO VYPRACOVÁNÍ:

Vytvořte EZS systém s možností rozšíření o prvky domácí automatizace. Systém bude využívat Raspberry Pi. Pro ovládání a konfiguraci systému se bude využívat dotykový display. Pro připojení čidel bude systém využívat technologii IEEE 802.15.4 nebo Bluetooth LE.

DOPORUČENÁ LITERATURA:

[1] VENKATESWARAN, Sreekrishnan. Essential Linux device drivers. Upper Saddle River: Prentice Hall, c2008,714 s. ISBN 978-0-132-39655-4

[2] KHAN, Ashfaq A. Practical Linux programming: device drivers, embedded systems and the Internet. Hingham: Charles River Media, c2002, xv, 420 s. ISBN 1-58450-096-4.

Termín zadání: 5.2.2018 ***Termín odevzdání:*** 21.5.2018

Vedoucí práce: Ing. Ondřej Krajsa, Ph.D.
Konzultant:

prof. Ing. Jiří Mišurec, CSc.
předseda oborové rady

ABSTRAKT

Diplomová práce se zabývá realizováním vlastního návrhu elektronického zabezpečovacího systému s možností rozšíření o prvky domácí automatizace. Ústředna je vytvořena pomocí Raspberry Pi3 B+ s dotykovým displejem, které komunikuje s čidly pomocí Bluetooth Low Energy.

KLÍČOVÁ SLOVA

Raspberry Pi, MQTT, Bluetooth Low Energy, Internet věcí, Node-Red, Elektronický zabezpečovací systém

ABSTRACT

The diploma thesis deals with the implementation of own design of an electronic security system with the possibility of extension with the elements of home automation. The control panel is built using the Raspberry Pi3 B+ with a touch screen that communicates with the sensors using Bluetooth Low Energy technology.

KEYWORDS

Raspberry Pi, MQTT, Bluetooth Low Energy, Internet of Things, Node-Red, Electronic security system

KUBŮ, Jiří. *Elektronický zabezpečovací systém s prvky IoT*. Brno, 2018, 62 s. Diplomová práce. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací. Vedoucí práce: Ing. Ondřej Krajsa, Ph.D.

PROHLÁŠENÍ

Prohlašuji, že svou diplomovou práci na téma „Elektronický zabezpečovací systém s prvky IoT“ jsem vypracoval(a) samostatně pod vedením vedoucího diplomové 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(ka) uvedené diplomové práce dále prohlašuji, že v souvislosti s vytvořením této diplomové práce jsem neporušil(a) autorská práva třetích osob, zejména jsem nezasáhl(a) nedovoleným způsobem do cizích autorských práv osobnostních a/nebo majetkových a jsem si plně vědom(a) 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(-ky)

PODĚKOVÁNÍ

Rád bych poděkoval vedoucímu diplomové práce panu Ing.Ondřeji Krajsovi, Ph.D. za odborné vedení, konzultace, trpělivost a podnětné návrhy k práci.

Brno ....

podpis autora(-ky)

Faculty of Electrical Engineering
and Communication
Brno University of Technology
Purkynova 118, CZ-61200 Brno
Czech Republic

http://www.six.feec.vutbr.cz

PODĚKOVÁNÍ

Výzkum popsaný v této diplomové práci byl realizován v laboratořích podpořených z projektu SIX; registrační číslo CZ.1.05/2.1.00/03.0072, operační program Výzkum a vývoj pro inovace.

Brno ....

podpis autora(-ky)

EVROPSKÁ UNIE
EVROPSKÝ FOND PRO REGIONÁLNÍ ROZVOJ
INVESTICE DO VAŠÍ BUDOUCNOSTI

OBSAH

Úvod **12**

1 Internet of Things - IoT **13**
1.1 Historie IoT . . . 13
1.2 Využití IoT . . . 13
1.3 Architektura IoT . . . 15
1.4 Protokoly IoT . . . 15
1.4.1 MQTT . . . 15
1.4.2 CoAP . . . 17
1.4.3 WebSocket . . . 18

2 Bluetooth **19**
2.1 Kompatibita verzí bluetooth . . . 19
2.2 BLE . . . 20
2.2.1 Topologie . . . 20
2.2.2 Stack BLE . . . 21

3 Raspberry Pi 3 **23**
3.1 Technické parametry . . . 23
3.1.1 Raspberry Pi3 modelB . . . 23
3.1.2 Raspberry Pi3 modelB+ . . . 24
3.2 Oživení Raspberry Pi . . . 24
3.2.1 Paměťová karta . . . 24
3.2.2 Výběr OS . . . 25
3.2.3 Vytvoření bootovací paměťové karty . . . 25
3.2.4 Spuštění raspbianu . . . 26

4 Node-Red **27**
4.1 Popis funkcí základních bloků . . . 29
4.2 Využití Node-Redu . . . 30
4.3 Ukázka práce v Node-Redu . . . 31
4.4 Zabezpečení . . . 35
4.5 Bluemix-IBM . . . 36
4.6 MQTT - HiveMQ . . . 37

5 Elektronický zabezpečovací systém **39**
5.1 Ústředna EZS systému . . . 39
5.1.1 EZS část . . . 40

5.1.2 Automatizační část 42
5.1.3 Informační část 45
5.1.4 EZS čidla 46
5.1.5 Python skript 47
5.1.6 Princip vytvořené komunikace - shrnutí 54
5.1.7 Možné rozšíření systému 56

6 Závěr **58**

Literatura **59**

Seznam symbolů, veličin a zkratek **61**

Seznam příloh **62**

SEZNAM OBRÁZKŮ

1.1 Počet bilionů IoT zařízení [10] . . . 14
1.2 Princip MQTT . . . 16
1.3 Architektura CoAP . . . 17
2.1 Kompatibilita BLE - převzato z:[8] . . . 19
2.2 Stack BLE - převzato z:[14] . . . 21
2.3 Packet na linkové vrstvě . . . 22
2.4 Komunikace BLE . . . 22
3.1 Raspberry Pi, displej - převzato z: [13] . . . 23
3.2 Prostředí Win32DiskImager . . . 25
3.3 Menu základního nastavení . . . 26
4.1 Prostředí Node-Red . . . 27
4.2 Manage Palette . . . 28
4.3 Bloky Node-Red . . . 29
4.4 Spárování BLE zařízení . . . 30
4.5 Naskenované zařízení BLE . . . 30
4.6 Ukázka programu . . . 31
4.7 Programování funkčního bloku . . . 32
4.8 Generace náhodných čísel - blok RANDOM . . . 32
4.9 Zaslání autorizač. e-mailu . . . 33
4.10 Změna hesla - uživatelské prostředí . . . 33
4.11 Ukázka Node-Red čidlo . . . 34
4.12 Node-Red zabezpečení . . . 35
4.13 Hash . . . 35
4.14 Nastavení přihlašovacího účtu . . . 36
4.15 IoT-IBM Bluemix . . . 37
4.16 Ukázka HiveMQ . . . 38
4.17 Node-Red příjem MQTT . . . 38
5.1 Funkční schéma EZS . . . 39
5.2 Grafické rozhraní . . . 40
5.3 EZS část . . . 41
5.4 EZS část hesla . . . 42
5.5 Automatizace . . . 43
5.6 BigTimer - část nastavení . . . 43
5.7 Ukázka termostat Node-Red . . . 44
5.8 Automatizace . . . 44
5.9 Informační část . . . 45
5.10 Informační část2 . . . 46

5.11 LaunchPad CC2640R2 46
5.12 Příjem dat Android aplikace 47
5.13 Status bluetooth 49
5.14 Datagram vytvořeného systému 55
5.15 Vytvořená struktura dat 55

SEZNAM TABULEK

1.1 Příklad bezdrátových technologií využívaných v IoT 15

ÚVOD

V dnešní moderní době se technika dostává do všech odvětví každodenního života nás všech. Obklopuje nás ze všech stran, stále více ji využíváme například i v domácím prostředí. Chytrá zařízení a automatizace zmíněného domácího prostředí je trendem této doby a bude se nadále neúprosně rozšiřovat.

Jednou z velmi důležitých věcí je i ochrana majetku a lidského zdraví. Diplomová práce je zaměřena na návrh elektronického zabezpečovacího systému s možností rozšíření o IoT prvky, především pro oblast domácí automatizace. K ovládání a konfiguraci celého systému je využíván především mikropočítač Raspberry Pi 3 od Britské firmy Premier Farnell. Bezdrátovou komunikaci mezi ústřednou a čidly zajišťuje Bluetooth Low Energy.

Hlavním cílem této práce je proniknutí do problematiky IoT technologie a realizování elektronického zabezpečovacího systému dle vlastního návrhu. Vytvořený systém je rozšířitelný o prvky domácí automatizace.

1 INTERNET OF THINGS - IOT

IoT – Internet of Things, v českém překladu internet věcí, je pojem, který označuje propojení fyzických objektů do sítě za účelem získávání dat, které jsou následně zpracovány. Jak již samotný název napovídá, v tomto případě se nejedená o jinou sít než Internet. Pomocí Internetu mají dané objekty mnohem větší možnosti počínaje komunikací mezi sebou, nebo například využívají již širokou škálu internetových služeb.

1.1 Historie IoT

Internet věcí není zase tak dlouho známý pojem. Poprvé se údajně tento termín objevil u jednoho ze zakladatelů výzkumné skupiny Auto-ID Labs. Jméno K. Ashtona můžeme tedy považovat jako hlavního průkopníka IoT. První oficiální informace o daném tématu prezentoval pan Asthon v roce 1999 a od té doby se tento trend neuvěřitelnou rychlostí rozrůstá. [15]

1.2 Využití IoT

Jak již bylo řečeno, tato technologie se stala velice rychle moderním trendem a neustále se rozrůstá do všech oblastí dnešního moderního světa. Nalezneme ji prakticky všude kolem nás. Ať už nahlédneme do průmyslových automatizací nebo domácího prostředí, kde může zajišťovat například monitoring zdraví nemocného člena rodiny. Využívání IoT by se nechalo pomyslně rozdělit do několika následujících oblastí.

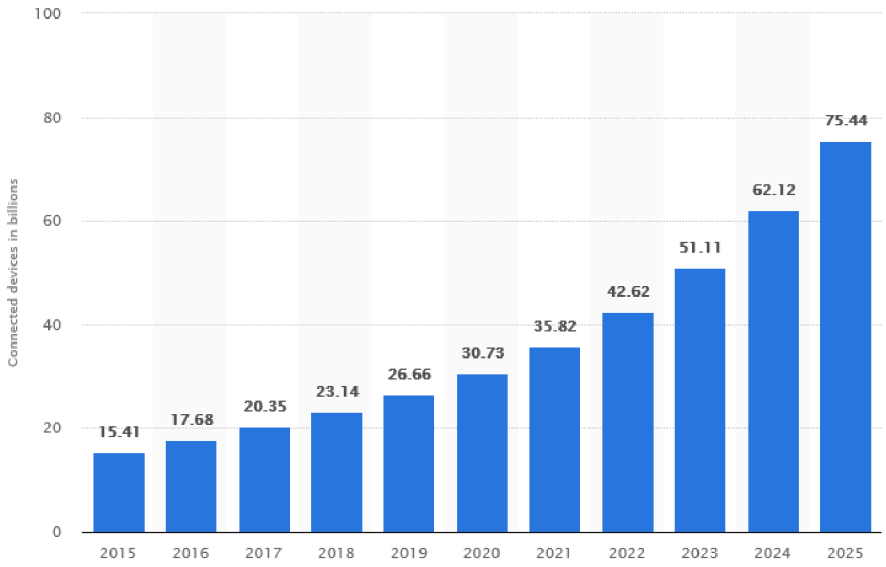
- Domácí automatizace – Home automation
- Průmyslová automatizace – Industrial Control
- Chytrá města – Smart Cities
- Chytré měřiče – Smart Metering
- Zdravotnictví a ochrana zdraví jednotlivců – E-Health
- Zemědělství – Smart Agriculture

Tato práce se zaměřuje především na první bod výše uvedených oblastí. Jedná se tedy o menší domácí sít, která bude sloužit především jako nástroj sběru dat z informačních čidel pro elektronický zabezpečovací systém. Tento navržený systém je možné rozšířit o prvky domácí automatizace jako je například dálkové ovládaní osvětlení, spínání topení či získávání dat o spotřebě elektrické energie.

Průmyslová automatizace, zde je nespočet získávaných dat takže o uplatnění IoT v tomto odvětví není pochyb. V posledních letech se hodně hovoří o chytrých městech. Jedná se o plošnou aplikaci této technologie na části města a zajistit tím plynulejší a bezpečnější chod města. Svých uplatnění zde IoT nalezne nespočet, příkladem

jsou propracované parkovací systémy apod. Jelikož většina rádiových systémů IoT jsou navrženy tak, aby byly málo energeticky náročné, nabízí se možnost trvalého umístění senzorů na místa bez servisní možnosti jako jsou například zmíněné parkovací plochy. Čidla by byla umístěna ve vozovce parkovacích míst a řidiči hledající místa na rozlehlých, přeplněných parkovištích budou přesně naváděni vytvořeným naváděcím systémem.

Chytré měřiče, co si pod tímto pojmem představit. Jedná se o měřící zařízení, které nepotřebuje pro přenos informací žádný obslužný personál. Získané data odesílá na stanovené uložiště, kde můžou být zpracovány v reálném čase. Umožní tak okamžitou informovanost o daném systému případně jeho rozšíření na základě přijímaných dat. Jedním zástupcem využívající Smart Metering je například technologie PLC (Power Line Comunication), která pro přenos dat využívá silnoproudé elektrické rozvody. Umožňuje tak zasílání dat, například z elektroměru, bez nutnosti budování dalších komunikačních kanálů v nepřípustných odstíněných místech. O velkém rozvoji této technologie vypovídá i následující statistický graf 1.1 připojených IoT zařízení.



Obr. 1.1: Počet bilionů IoT zařízení [10]

.

1.3 Architektura IoT

Z určitého pohledu si celou technologii můžeme zjednodušeně představit jako tři bloky, které spadají do dané architektury.

Prvním prvkem jsou Things, tedy v překladu věci. Jedná se vlastně o jakékoliv zařízení, které je připojeno do nějaké širší sítě, například Internetu. Připojení může být realizováno dvěma základními způsoby. Jak se dá předpokládat jedná se o drátové spojení, například pomocí vodičových sběrnic, nebo bezdrátového přenosu jako je technologie Bluetooth.

Druhým prvkem není nic jiného než samotná komunikační síť nebo brány zajišťující spojení několika věcí do jednoho cloudu.

Třetí prvek může být uložen například někde v datových centrech. Jedná se tedy o vzdálené servery (cloudy), které mají za úkol bezpečně ukládat získaná či přenesená data.[16]

1.4 Protokoly IoT

Jedním z nejrozšířenějších protokolů : WPAN (Wireless Personal Area Network) se stal Bluetooth Low Energy (BLE), jinak přezdívaný Smart. Z hlediska nízké energetické náročnosti systému má zde svou prioritu. Následující tabulka 1.1 zobrazuje základní parametry nejrozšířenějších technologií využívající BLE.

Tab. 1.1: Příklad bezdrátových technologií využívaných v IoT

Protokol	Pásmo	Dosah	Přenosová rychlost
Bluetooth	2,4GHz	100m	100 kbit/s (BLE)
ZigBee	2,4GHz	10-100m	20-250 kbit/s
Z-Wave	868MHz(EU)	30m	250 kbit/s
SigFox	867MHz(EU)	3-50km	až 1 kbit/s
LoRaWAN	867MHz(EU)	2-22km	až 0,3-50 kbit/s

1.4.1 MQTT

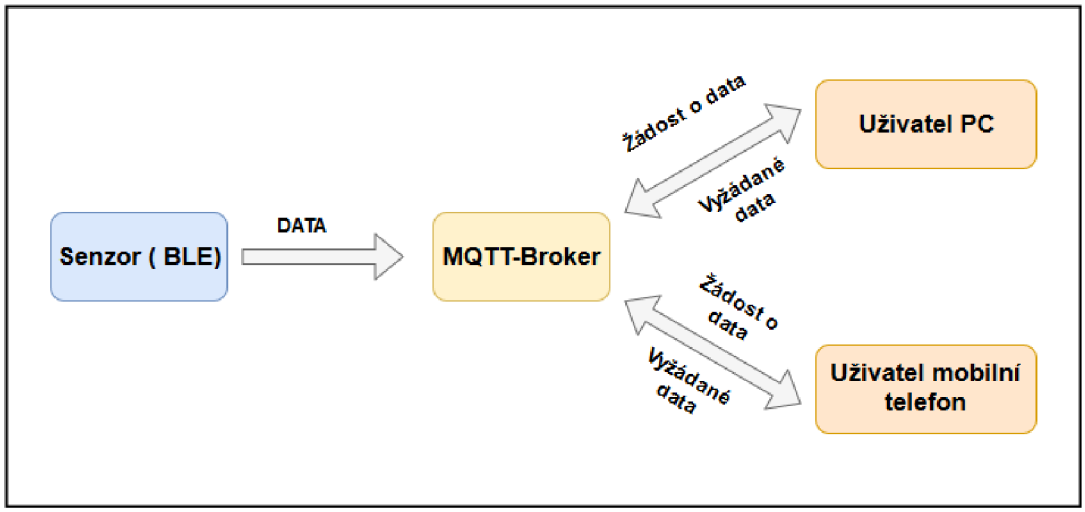
V celém znění Message Telemetry Transport. Jedná se o komunikační protokol, který pracuje nad protokolem TCP/IP. Byl navržen především pro místa, kde je malá propustnost sítě či vysoká odezva. Dá se tedy říci, že tento protokol je určen pro né moc spolehlivé sítě. Standard definuje tři úrovně spolehlivosti při doručování zpráv. Jedná se o služby QoS, což je jedním z hlavních parametrů MQTT. Nastavení QoS se dělí na dvě části. První částí je od zdroje dat k brokeru a druhou částí z brokeru

k příjemci. Nastavení QoS směrem k brokeru probíhá na každé publikované zprávě. Druhý případ, tedy přijímání zpráv si stanoví klient při přihlašování k odběru dat. [5]

Zde se QoS dělí do tří skupin:

- Přesně jednou - jedná se tedy o spolehlivé doručení zprávy. Přijímač zaznamená jen jednu zprávu a několik pokusů o doručení. Využití nalezne tam kde by duplicita mohla způsobit chyby.
- Nejméně jednou – v tomto případě může docházet k duplicitám na straně přijímače.
- Nejvýš jednou – případné nedoručení dat se neřeší. Tato aplikace je vhodná například při periodickém odesílání informačních dat kdy jeden „vzorek“ z desítek vzorků za sekundu nemá velký význam.[2]

Na následujícím obrázku 1.2 nalezneme schéma vysvětlující princip MQTT.



Obr. 1.2: Princip MQTT

Mosquitto

Existuje mnoho implementací broker. Jedním z nejznámějších je broker s názvem Mosquitto. Jedná se o open-source MQTT broker, který podporuje všechny nejznámější programovací jazyky. Umožňuje tedy komunikaci mezi větším množstvím zařízení. Využívá k tomu samozřejmě MQTT protokol. Zařízení můžeme rozdělit do dvou režimů:

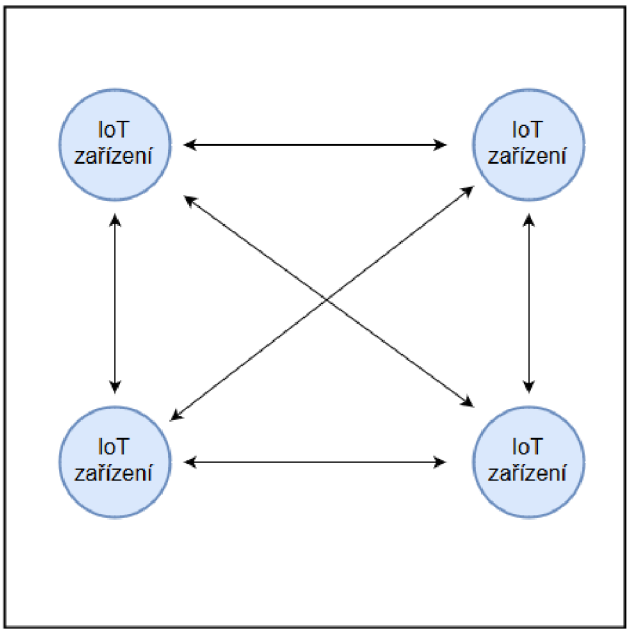
- Publisher - zařízení generující data a ukládá do front
- Subscriber - načítá data z fronty od publishera

Když to shrneme, princip je velmi jednoduchý. Nejprve jsou nějakým IoT čidlem generovány data, následně vloženy do MQTT. Dalším krokem je vyčítání MQTT dat k čemu může sloužit například Node-Red ve kterém se nastaví další operace s daty. [9]

Instalace Mosquitto serveru je poměrně jednoduchou záležitostí a lze jí provozovat jak na Linuxu tak i na Windows.

1.4.2 CoAP

Constrained Application Protocol (CoAP) je webový protokol pracující na aplikační vrstvě ISO/OSI. Tato komunikace je určena primárně pro machine-to-machine (M2M). Využívá protokol UDP, který je nespolehlivý a není tedy zajištěna doručitelnost všech paketů či jejich správného pořadí. Samotné zajištění přenosu probíhá pomocí asynchronního přenosu zpráv.[3]



Obr. 1.3: Architektura CoAP

.

Využívá klasického modelu dotaz/odpověď. Výhodou užívání UDP protokolu je možnost realizace IP multicastu pro skupinovou komunikaci. Naopak nevýhodou využívání UDP je absence využívání zabezpečení. Například nelze použít protokoly TLS/SSL. [4] Zabezpečení proti odposlechu a dalším způsobům získání dat třetí stranou, nebo jen chyby způsobené přenosem, lze řešit při využívání UDP protokolu i jiným způsobem. Například pomocí DTLS, které je podobné TLS. Řeší i případné ztráty paketů, jelikož veškeré pakety jsou očíslovány a v případě chyby jsou vyžádány k opětovnému zaslání.

I když protokol CoAP není na takové úrovni jako například MQTT, stále se vyvíjí což může být v rámci implementace i nežádoucí. Ale s výhodou dotaz/odpověď může být jednou na stejné úrovni jako například MQTT.

M2M

Jedná se o komunikaci mezi přístroji dle samotného názvu Machine to Machine, která je základem celého Internetu věcí. Základní princip je navržen mezi dvěmi a více přístroji tak aby nebylo potřeba přímé interakce člověka. Koncovým prvkem není tedy člověk jako je tomu u H2H sítě. Samotný proces přenosu dat může být realizován drátovou či bezdrátovou technologií.

Sítě M2M jsou dnes velmi rozšířené především v průmyslové oblasti kde zefektivňují administrativní provoz, čímž samozřejmě šetří nemalé finanční prostředky. Průkopníkem této sítě byl v roce 1968 T.G.Paraskevakos který začal implementovat dnešní M2M na „chytrém“ telefonu. Jednalo se o technologii Calleer-ID jehož hlavní myšlenka je zasílánání udajů o volaném společně s hlasovými daty. Tehda se samozřejmě jednalo o zasílání dat po obyčejné telefonní lince.[6]

1.4.3 WebSocket

Jedná se o protokol jehož hlavní možností je obousměrná komunikace mezi jednotlivými TCP schránkami. Hlavní princip je takový, že je vytvořeno se serverem jedno spojení pomocí TCP, které je neustále navázané. V případě potřeby webových aplikací není zapotřebí neustále navazovat nově spojení. Pracuje na portech 80 a 433. [11]

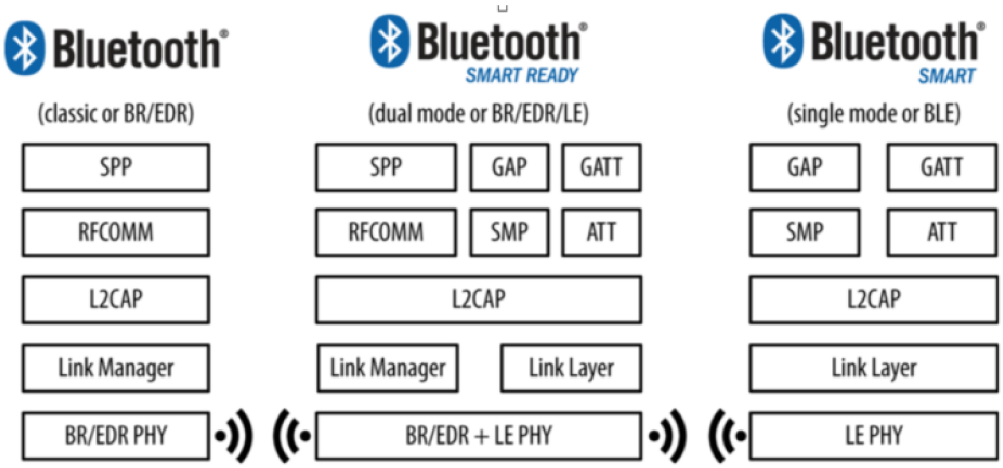
2 BLUETOOTH

Jedná se o technologii, která je globálním standardem bezdrátové komunikace s poměrně dlouhou historií, kdy stihla projít významnou úpravou. Přelom nastal s verzemi od 4.0, které se nazývají Bluetooth low energy a jsou označovány zkratkou BLE případně Bluetooth Smart.

První kdo odstartoval převratný pokrok Bluetooth technologie[7], za účelem snížení spotřeby energie na mnohonásobně nižší hodnoty, byla firma Nokia. Rok 2010 byl významný pro skupinu SIG (Bluetooth Special Interest Group), kdy si tento standard koupila a nadále pracovala na jeho dalším vývoji. Jelikož hlavní předností BLE je ohled na nízkou energetickou náročnost[8], byl předurčen pro rozvíjející se odvětví IoT. Dnes zařízení komunikující pomocí BLE, vykazují velice dlouho životnost a to mnohdy i s minimálními zdroji energie. V poslední době se objevují zajímavé návrhy řešící problém s napájení BLE zařízení. Hlavní myšlenkou těchto návrhů je, že by BLE zařízení „recyklovalo energetický šum“ z okolí. BLE zařízení by se tedy sama dobíjela například z hodně rozšířených WiFi sití, které jsou téměř všude kolem nás.

2.1 Kompatibita verzí bluetooth

I přes to, že starší verze využívají stejné rádiové frekvence jako novější verze bluetooth, nejsou spolu kompatibilní. Jediná přechodná verze 4.0 umožňovala komunikaci jak se starší tak i novější verzí. Následující obrázek 2.1 vypovídá a kompatibilitě verzí bluetooth. Levý sloupec popisuje klasické starší bluetooth, prostřední sloupec je přechodná verze a nakonec pravý sloupec zobrazující BLE.[8]



Obr. 2.1: Kompatibilita BLE - převzato z:[8]

.

2.2 BLE

Bluetooth Low Energy vznikl za účelem nového komunikačního protokolu bezdrátové technologie s hlavní myšlenkou snížení energetické náročnosti oproti starším verzím bluetooth. Pokles spotřeby energie byl zajištěn především zkrácením doby aktivního využití rádia při komunikaci, čímž bylo zajištěno zbytečnému plýtvání energie.

2.2.1 Topologie

BLE technologii můžeme rozdělit na dvě části. Prvním je vysílání všesměrové a druhým takzvané vysílání connection, tedy s navázáním přímého spojení.

Všesměrové

K tomuto druhu vysílání jsou zapotřebí dva typy zařízení. Jedním je broadcaster (vysílač) a druhý observer (přijímač). Vysílač v tomto případě nemá zájem o přijímání dat a slouží tedy jen k vysílání dat. (vysílá takzvané advertising pakety). To, že nenavazuje spojení jen s jedním přijímačem má i své výhody. Vysílaná data může přijímat více různých přijímačů zároveň. Všesměrový paket ve standardní velikosti má 31 b což má značný vliv na maximální přenos dat, ale o to je celý systém jednodušší a rychlejší. Své uplatnění tedy nalezne především u zasílání dat s omezenou velikostí pro více přijímacích zařízení. Jelikož se jedná o jednoduché zasílání dat komukoliv je úplně bez zabezpečení. Citlivá data by tedy mohl kdokoliv odposlechnout.[8]

Connection

Tato topologie je standardem Bluetooth low energy. Využívá se především tam kde je potřeba obousměrná komunikace nebo když je všesměrový paket nedostačující z pohledu velikosti přenášených dat.

Po navázání spojení je neustále udržována komunikace periodickou výměnou paketů. Pro tento případ je zapotřebí aby se určili dvě základní role v topologii.

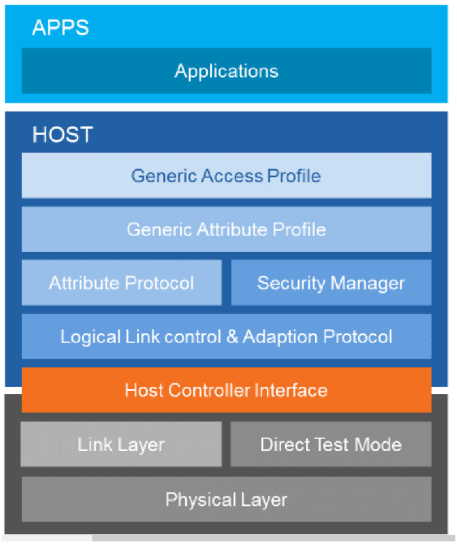
Master – jedná se o centrálu topologie, opakovaně skenuje stanovené frekvence a snaží se vyhledat advertising (všesměrový) paket. V případě, že paket nalezne, naváže spojení a vysílač přestane vysílat všesměrové pakety k navázání spojení. Vysílač celou svou komunikaci soustředí pouze na Master bez vyhledávání advertisingu.

Slave – jedná se o zařízení vysílající, v určitých intervalech, všesměrové pakety pro navázání komunikace.

V tomto případě lze tedy vytvořit topologii, kterou lze nějákým způsobem organizovaná oproti všesměrové komunikaci kde všechny přijímače v dosahu získaly vysílaná data. Při vytváření topologií pomocí Connection lze u novějších verzích Bluetooth využívat možnost kdy se jedno zařízení tváří současně jako master i slave.[8]

2.2.2 Stack BLE

Podíváme-li se na BLE komunikaci z pohledu sítových modelů, rozdělíme ji do jednotlivých vrstev. V globálu se takovýto referenční model nazývá stack a zde je reprezentován následujícím obrázkem 2.2 V následujícím odstavci jsou popsány dvě nejdůležitější části stacku.



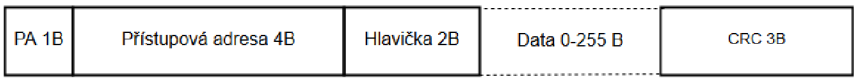
Obr. 2.2: Stack BLE - převzato z:[14]

.

Fyzická vrstva – zde komunikace probíhá v bezdrátovém frekvenčním pásmu 2,4GHz. Jelikož se jedná o nelicencované frekvenční pásmo, je využíváno společně s dalšími technologiemi jako je například Wi-Fi. Pro přenos dat byla stanovena modulace GFSK. Jedná se o druh klíčování založené na principu dvou frekvencí přičemž je jedna frekvence přiřazena logické nule a druhá naopak logické 1. Celé frekvenční pásmo je rozděleno na 40 kanálů s rozestupy o velikosti 2MHz . Samotné odlišení logické 0 a 1 je stanovené dle změny frekvence vůči střední fekvence každého kanálu. Přesněji řečeno se jedná o 185 kHz výše či níže od střední frekvence.

Linková vrstva – základní vlastností této vrstvy je velice přesná práce v reálném čase. V první řadě tato vrstva pracuje se všemi 40 kanály z čehož jsou 3 kanály využívány pro „zvláštní“ účely. Přesněji řečeno se jedná o kanály které slouží k vysílání advertisingových paketů které se využívají k navázání spojení. Tyto pakety mohou obsahovat omezené množství dat které lze vysílat. Ostatních 37 kanálů jsou určeny pro komunikaci po navázání spojení. Jak je vidět na následujícím obrázku 2.3, datový paket je zasílán uvedeném stavu. [1] Preambule 1B, přístupové adresy 4B, samotná hlavička má 2B a jelikož maximální velikost dat je 255B, určuje typ dat a jejich délku. Na závěr kontrolní součet CRC, který je vymezen 3B. Pro výše po-

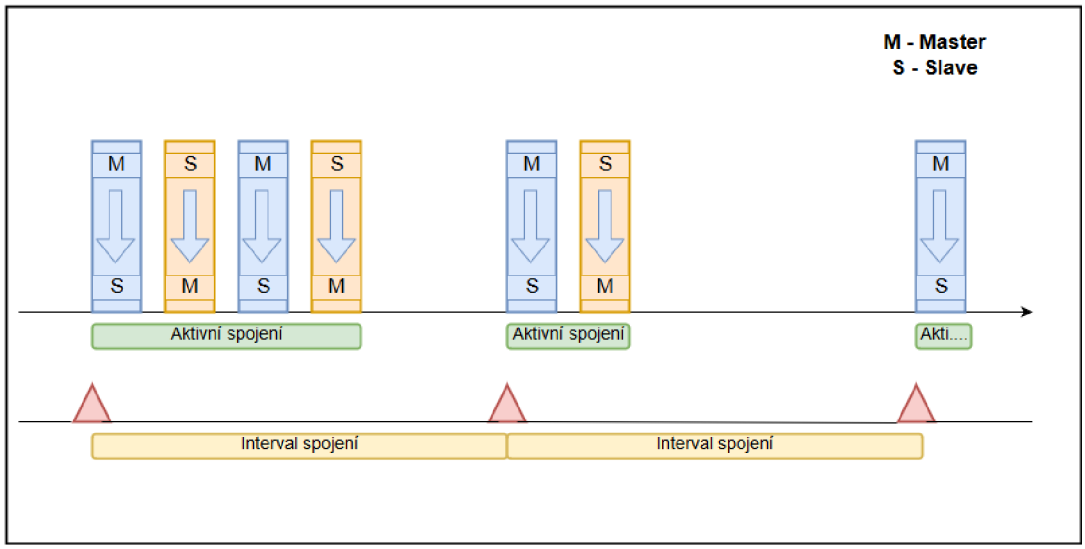
psanou skupinu 3 kanálů je vytvořen trochu odlišná struktura paketu, kdy hlavním rozdílem je omezená velikost datové části paketu na 37 B. [17]



Obr. 2.3: Packet na linkové vrstvě

.

Celá komunikace je složena z intervalů, které se periodicky opakují a umožňují tak vytvoření mnohem efektivnější komunikace z pohledu energetické náročnosti. Princip je poměrně jednoduchý, na začátku intervalu dojde k vzájemné komunikaci mezi zařízeními a po přenesení dat je rádio v daném intervalu vypnuté. Opětovně se zapne až na začátek dalšího intervalu. Tento proces komunikace můžeme vidět na následujícím obrázku 2.4, který naznačuje intervalovou komunikaci.

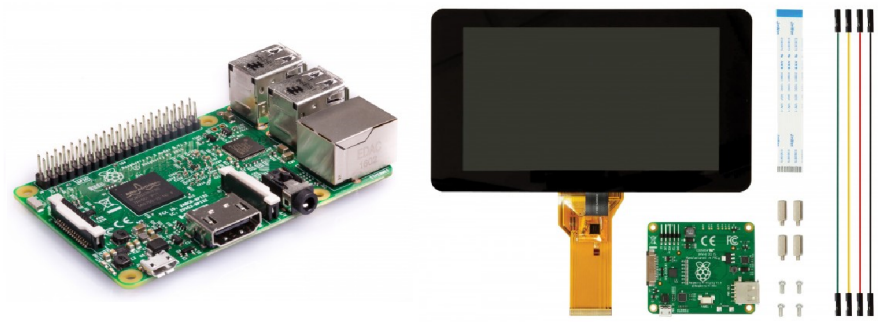


Obr. 2.4: Komunikace BLE

.

3 RASPBERRY PI 3

Jedná se o jednodeskový mikropočítač s nízkou spotřebou, max. 3,5 W založený na ARM architektuře. První model Raspberry Pi se objevil na trhu v roce 2012 a ihned si získal své příznivce. Za pouhý rok bylo prodáno 2 miliony kusů což umožnilo rychlý vývoj dalších verzí. Pro realizaci této práce bylo vybráno Raspberry Pi 3 model B, který vidíme v levé části následujícího obrázku 3.1 V průběhu vytváření práce se začala projevovat závada, pravděpodobně na integrovaném přijímači Bluetooth, proto bylo zakoupeno nové Raspberry Pi, kde se předcházející problémy neprojevily. Jelikož se mezitím na trh dostala nová verze Raspberry Pi3 model B+, byl pro pokračování této práce zvolen zmíněný aktuálnější model, které je výkonnější.



Obr. 3.1: Raspberry Pi, displej - převzato z: [13]

.

3.1 Technické parametry

Tyto dva nejaktuálnější modely jsou nejvýkonějším, ale také nejdražším, mikropočítačem řady Raspberry. Pořizovací cena každého z nich se pohybuje kolem 1000 Kč bez příslušenství.

3.1.1 Raspberry Pi3 modelB

Na dalších řádcích nalezneme hlavní parametry modelu B:

- Architektura - ARMv8-A (64/32-bit)
- Procesor (CPU) - 1.2 GHz 64-bit quad-core ARM Cortex-A53
- Paměť (SDRAM) - 1 GB (sdílená s GPU)
- USB 2.0 porty - 4 (přes zabudovaný pětiportový USB hub; jeden USB port vnitřně propojen s ethernet portem)
- Video vstup - 15-pinový MIPI konektor kamerového rozhraní (CSI)

- Video výstup - HDMI (rev 1.3 , 1.4), 14 HDMI rozlišení od 640×350 do 1920×1200
- Interní paměť - MicroSDHC, USB Boot Mode
- Integrovaná síť - 10/100 Mbit/s Ethernet + WiFi 802.11n a Bluetooth 4.1 [13]

3.1.2 Raspberry Pi3 modelB+

Jedná se o podstatně vylepšenou a zrychlenou modelu 3B. Především v oblasti integrované sítě. Parametry této nové verze jsou následující:

- Architektura - ARMv8-A (64/32-bit)
- Procesor (CPU) - 1.4 GHz 64-bit quad-core ARM Cortex-A53
- Paměť (SDRAM) - 1 GB (sdílená s GPU)
- USB 2.0 porty - 4 (přes zabudovaný pětiportový USB hub; jeden USB port vnitřně propojen s ethernet portem)
- Video vstup - 15-pinový MIPI konektor kamerového rozhraní (CSI)
- Video výstup - HDMI (rev 1.3 , 1.4), 14 HDMI rozlišení od 640×350 do 1920×1200
- Interní paměť - MicroSDHC, USB Boot Mode
- Integrovaná síť - Gigabitový Ethernet skrze USB 2.0 + WiFi 802.11ac, Bluetooth 4.2 BLE a možnost PoE napájení skrze externí HAT modul [13]

V této práci je dále využíván oficiální RPi kapacitně dotykový 7“ displej s rozlišením 800x480 DSI, který můžeme vidět v pravé části obrázku 3.1. Displej má plnou podporu Raspbianu, proto jeho instalace a propojení s Raspberry není náročná. Zvládá rozpoznat až 10 dotyků najednou a jeho pořizovací cena činí přibližně 2000 Kč.

3.2 Oživení Raspberry Pi

První kroky k úspěšnému použití Raspberry vedou přes instalaci operačního systému a Před samotnou prací s Raspberry je zapotřebí přípravy v podobě instalace operačního systému a základního nastavení, které je popsáno v této kapitole.

3.2.1 Paměťová karta

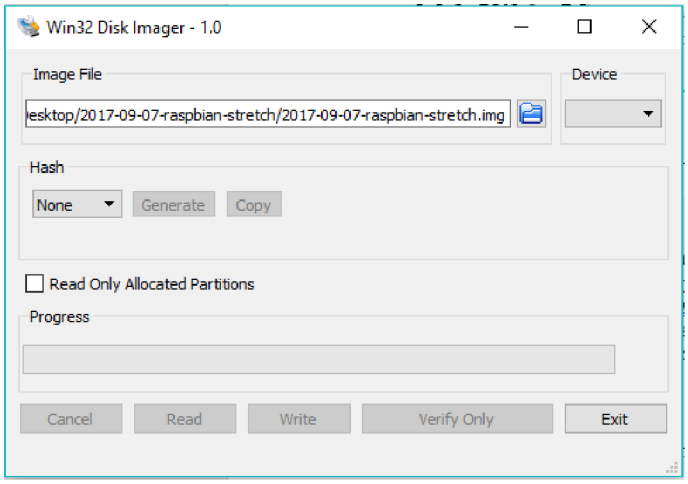
Pro přípravu OS pro Raspberry Pi bude zapotřebí v první řadě dostatečně velká paměťová karta s větší rychlostí zápisu a čtení dat. V této diplomové práci byl operační systém instalován na kartu Kingston microSDXC 64GB UHS-I která spadá do rychlostní třídy Class 10. Její minimální přenosová rychlost je 10 MB/s a maximální rychlostí četní/zápis až 90/45 MB/s.

3.2.2 Výběr OS

Jelikož se jedná o poměrně rozšířený mikropočítač je více možností volby operačního systému. Pro účely diplomové práce byl zvolen Raspbian, který lze zdarma stáhnou v aktuální verzi z následujících stránek: `https://www.raspberrypi.org/downloads/`. Mezi další nejznámější možnosti OS patří například Debian ARM, MATE či Ubuntu. Jak již sám název napovídá, Raspbian byl vytvořen přímo pro námi využívané Raspberry Pi. K dispozici je i v odlehčené verzi bez grafického prostředí takzvaný RASPBIAN STRETCH LITE. Pro tuto práci byla využita klasická verze OS RASPBIAN STRETCH WITH DESKTOP.

3.2.3 Vytvoření bootovací paměťové karty

Stažený balíček operačního systému nestačí pouze uložit na danou paměťovou kartu. V první řadě je zapotřebí vytvořit bootovací sekci pro spuštění systému v zařízení. Tento krok lze vytvořit více způsoby, jelikož existuje mnoho programů bohužel někdy až trochu pochybných. Proto asi nejspolehlivějším programem je Win32DiskImager, který je doporučován i v oficiálním návodu na Raspberry.



Obr. 3.2: Prostředí Win32DiskImager

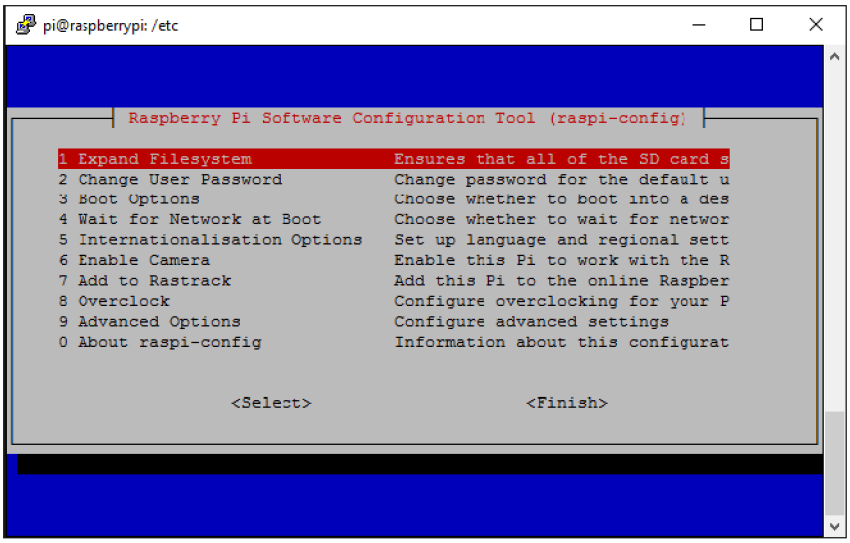
.

Win32DiskImager nalezneme ke stažení na stránkách `https://sourceforge.net/projects/win32diskimager/` , kde je k dispozici zdarma pro Windows, Linux i Mac. Jeho vizuální prostředí, obrázek 3.2, je velice jednoduché. V první řadě je potřebné zvolit .img obraz, který nalezneme po rozbalení staženého balíčku. Následně volba správné síťové jednotky, kam se po stisknutí tlačítka Write vše nakopíruje včetně bootovací sekce.

3.2.4 Spuštění raspbianu

Do Raspberry stačí vložit předinstalovanou kartu. Poté pomocí micro USB připojit zdroj napájení, který musí být minimálně 1,5A a systém se následně sám spustí. Pomocí připojeného HDMI monitoru provedeme nejprve základní nastavení povolení přístupu pomocí SSH.Tyto možnosti nalezneme v v hlavním menu ve správci nastavení. Dalším krokem je zjištění přidělené IP adresy zařízení. Nejjednodušší je použití příkazu cmd arp -a. Pro použití SSH protokolu je mnoho programů, v našem případě byl využit volně dostupný a velmi oblíbený program Putty. Po jeho spuštění stačí zadat zjištěnou IP adresu, nastavit port na 22, který SSH protokol využívá a po stisku open se naváže spojení.

Po připojení a přihlášení se pomocí přednastaveného uživatelského jména a hesla povedou první kroky do sekce „Change User Password“ (obrázek 3.3). Do menu software configuration tool se dostaneme pomocí příkazu sudo raspi-config a přenastavíme přístupové heslo. Zde můžeme nastavit i další parametry jako je například využívání celé paměťové karty pro potřeby RaspberryPi. V základním nastavení má systém vymezen jen určitou část.

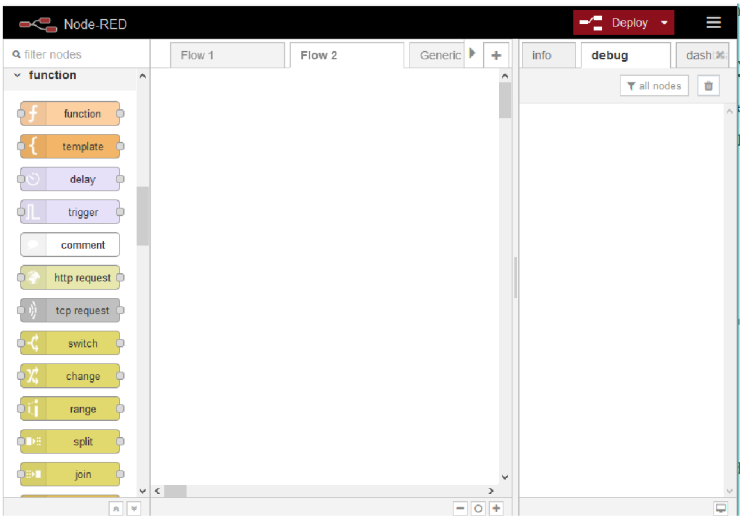


Obr. 3.3: Menu základního nastavení

.

4 NODE-RED

Node-RED je programovací nástroj, od firmy IBM, sloužící k propojení hardwarových zařízení s online službami. Dalo by se říct, že se jedná o programovací editor umožňující propojení datových toků. Prostředí samotného Node-Redu můžeme vidět na následujícím obrázku 4.1.



Obr. 4.1: Prostředí Node-Red

.

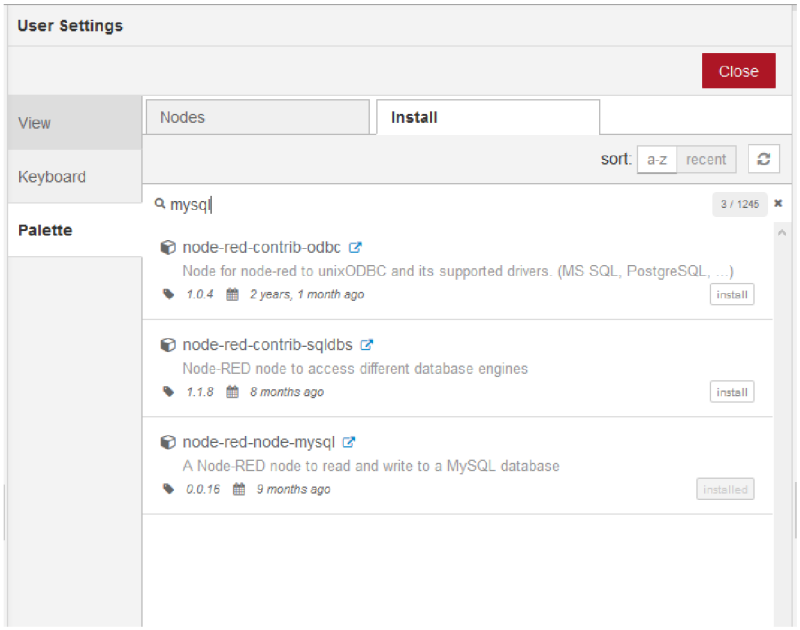
V levé části nalezneme menu s uzly (bloky), které lze rozšiřovat pomocí velikého výběru instalačních npm balíčků. Samotná instalace balíčků se provádí v menu Manage palette, obrázek 4.2, které umožňuje přidávat/odebírat či aktualizovat veškeré balíčky. Tuto správu lze provádět i pomocí příkazů v terminále. Samotnou instalaci mohou doprovázet chyby, které lze řešit ruční instalací přes terminál. Největší obtíže zde způsobují kolize starších verzí instalovaných balíčků. Mnohdy jen některé starší pozůstatky z upgradovaných verzí samotného Node-Redu. V takovém případě se ukázalo jako nejschůdnější řešení ruční prohledání adresářů a úplné smazání záznamů daných balíčků, případně starších verzí Node-Redu.

Jednotlivé uzly s definovanými funkcemi se přesunují na pracovní plochu, kterou nalezneme v prostřední části obrázku 4.1. Pracovní plochy jsou zde rozděleny do jednotlivých listů nazývaných Flow. Každý uzel je charakterizován výstižným názvem pro danou problematiku či funkci programu. Po jejich rozvržení na pracovní ploše se pomocí propojů pospojují dle vytvářené logiky. Po stisknutí tlačítka Deploy dojde na pozadí ke "sladění"jednotlivých předprogramovaných částí v jeden celkový provázaný program.

Další důležitou možností jsou nastavení funkce deploy. Provedení deploy kompletního projektu může být především ve větších aplikacích časově náročné. Proto

jsou zde na výběr možnosti provedení pouze částečného buildu programu, který se zaměří například jen na pozměněné flow případně řeší pouze samotné části programu, které byly změněny. Tímto způsobem dochází k časovým úsporám při práci v tomto prostředí. V pravé části obrázku nalezneme konzoly, do které lze nasměrovat definované výstupy. Zobrazují se zde i potřebné informace k případným chybám.

Poslední verze Raspbianu už mají implementovaný Node-Red přímo v sobě. Je proto důležité ověřit aktuálnost dané verze a případně ji aktualizovat. Po samotné instalaci a spuštění programu se na vytvořený server přistupuje pomocí lokální adresy `http://lokální_adresa:1880`. V případě grafického výstupu se výsledek zobrazí na adrese `http://lokálníadresa:1880/ui/`. Ideálním případem je zajištění veřejné IP adresy, aby celý projekt byl přístupni z Iternetu. V takovém případě je důležité zabývat se i otázkou zabezpečení.



Obr. 4.2: Manage Palette

Pro zajištění automatického startu Node-Redu se využívá příkaz, který nalezneme v následujícím výpisu. V této práci byl autostart potřebným, aby samotný uživatel EZS nemusel zasahovat do spouštění systému při případném výpadku zálohovaného zdroje napájení.

Výpis 4.1: Autostart Node-Red

```
1    sudo systemctl enable nodered.service
```

4.1 Popis funkcí základních bloků

Jak již bylo řečeno, balíčků s funkčními bloky je nepřeberné množství. Stačí navštívit oficiální stránky Node-Redu a pravděpodobnost nalezení potřebného bloku je vysoká. Odkaz na zmíněné webové stránky nalezneme pod následující adresou `https://nodered.org/`. Obrázek 4.3 prezentuje hlavní bloky (uzly) využívané při programování této práce. Po samotné instalaci Node-Redu je naimportován základ balíčků. Velká část z níže uvedených byla doinstalována pomocí manage palette, při případných instalačních potížích pomocí terminálu.

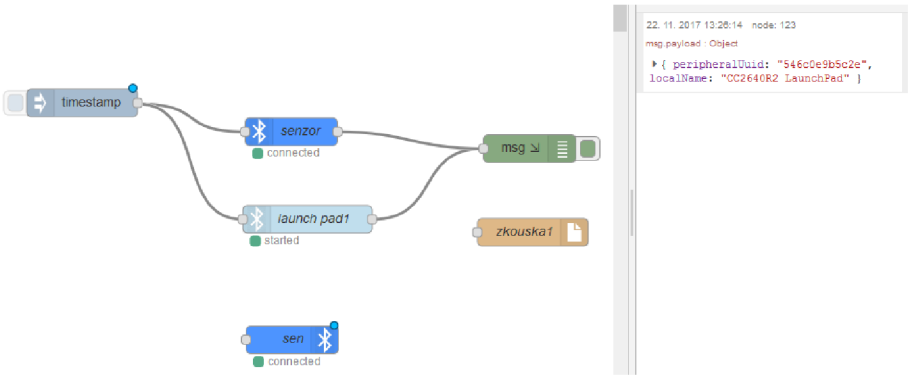
Obr. 4.3: Bloky Node-Red

Inject blok složí k automatickému odesílání zpráv s určitými parametry. Debag je výstupní blok směřující přijatá data do konzole, ve které můžeme zobrazit zasílaný výstup nebo případnou chybovou hlášku. MQTT blok zajišťuje správu MQTT komunikaci. Zajišťuje připojení na definovaný MQTT broker. Function blok složí k definici vlastního vlastního programu v jazyku JavaScript. V případě, že není nalezen blok s požadovanou funkcí, zde lze jeho vlastnosti nadefinován téměř bez omezení. Delay zajišťuje časovou prodlevu. Comment je popisový blok, jehož obsah daný program nezpracovává. Audio out je blok zajištující zvukový výstup. String slouží k práci s řetězci, umožňuje vyhledávání v řetězci, nebo k úpravě přijatého řetězce dle definovaných pravidel. Camerapi take photo je blok zajištující správu připojené kamery. Switch se rozhoduje na základě stanovených pravidel, kterým výstupem dané data nasměruje. Change je blok, který přijatá data zamění a odešle

na výstup. Sqlite zajišťuje databázi SQLite. Email blok slouží pro odeslání email adresátovi, případný příjem emailu. Bigtimer slouží k detailnímu naprogramování časového spínače. Button a Text jsou zástupci grafického rozhraní, jejichž význam je zřejmý. File slouží k práci se souborem.

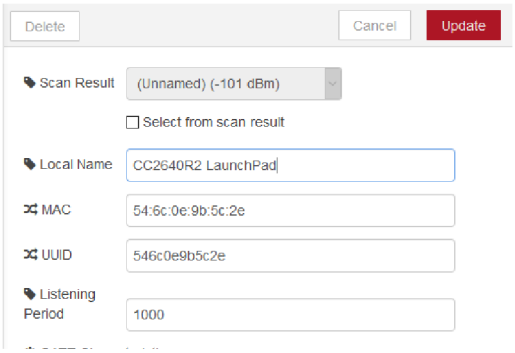
4.2 Využití Node-Redu

Větší část programování práce byla vytvořena právě v tomto programu. Mezi jeho hlavní výhodu patří usnadnění práce s programováním a jeho grafické zobrazení. Na následujícím obrázku 4.4 je zaznamenán prvotní pokus o komunikaci s čidlem. Tento způsob ověřování byl důležitým prvkem při hledání závady komunikace, která byla popsána v semestrální práci. Jednalo se o poměrně velkým problémem, jelikož se z neznámých důvodů komunikace mezi čidlem a Raspberry stávala velmi nestabilní. Jak už bylo jednou zmíněno, problém nebyl softwarového rázu, ale pravděpodobně byla závada přímo na BLE adaptéru na Raspberry.



Obr. 4.4: Spárování BLE zařízení

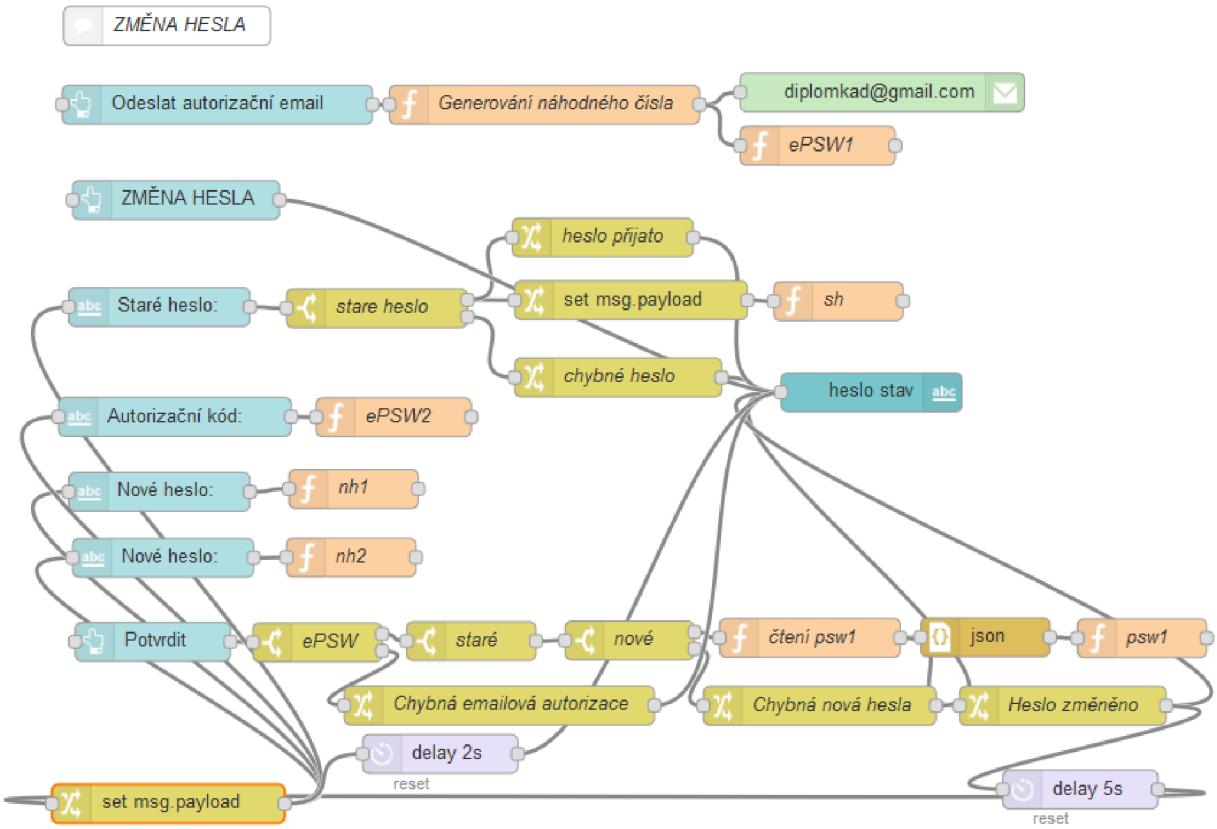
Na následujícím obrázku 4.5 je zobrazen výsledek skenování okolí. Bylo nalezeno BLE čidlo, jehož identifikační údaje jsou z obrázku patrné.



Obr. 4.5: Naskenované zařízení BLE

4.3 Ukázka práce v Node-Redu

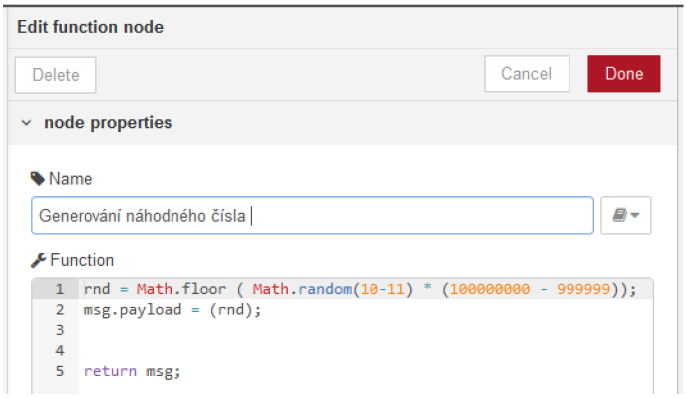
V následující části nalezneme praktickou ukázku práce s Node-Redem. Jedná se o část programu této práce. Konkrétně na obrázku 4.6 nalezneme řešení změny hesla ve vytvářené EZS.



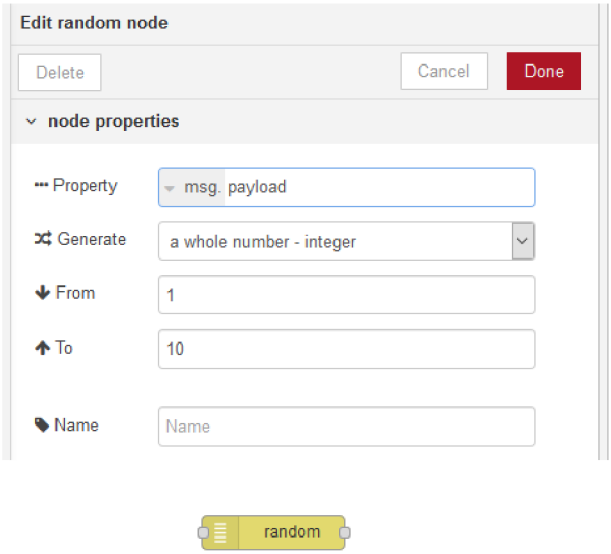
Obr. 4.6: Ukázka programu

V levé části obrázku 4.6 vidíme modré bloky, které prezentují propojení s vizuální částí programu, takzvaným dashboardem. Samotné funkce jednotlivých bloků nalezneme v kapitole 4.1. Zaměřme se na horní část obrázku kde nalezneme tlačítko na odeslání autorizačního emailu. Tato část je připojena na funkční blok, který má za úkol vygenerovat náhodné číslo o určité délce. Samotné prostředí editace tohoto funkčního bloku vidíme na obrázku 4.7, kde je naprogramovaná funkce pomocí JavaScriptu. Druhou možností, jak vyřešit vygenerování náhodného čísla, je nainstalování určitých balíčků, které obsahují blok random s předdefinovanými funkcemi. Podobu a jeho editační prostředí nalezneme na obrázku 4.8. Při porovnání těchto dvou možností můžeme vidět hlavní výhodu Node-Redu, která spočívá ve velké podpoře programu a nespočtu balíčků, které lze doinstalovat, případně si vytvořit zcela

vlastní.

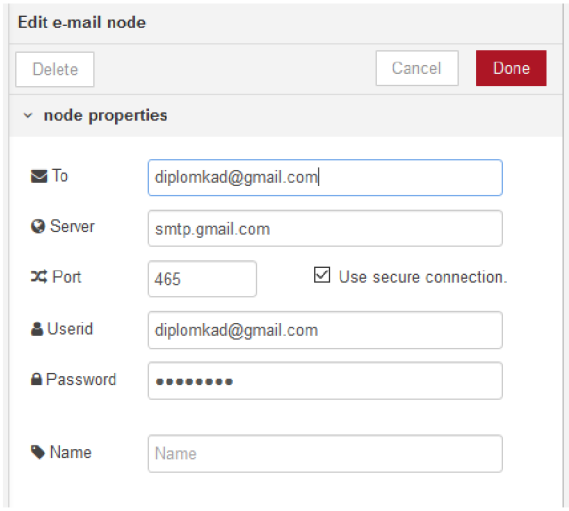


Obr. 4.7: Programování funkčního bloku



Obr. 4.8: Generace náhodných čísel - blok RANDOM

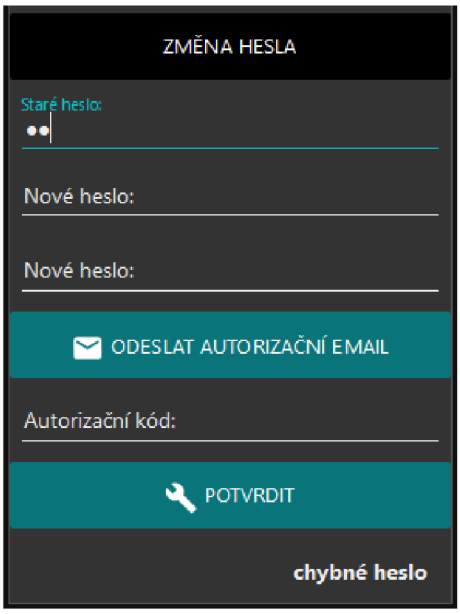
Jsme-li v grafickém prostředí a klikneme na tlačítko odeslání, generátor náhodného čísla vygeneruje číslo, které odešle na výstup. Jak můžeme dále vidět, na výstupu jsou připojeny dva bloky. První, zelený blok, zajišťuje zaslání vygenerovaného čísla na emailovou adresu správce systému. Nastavení tohoto bloku vidíme na obrázku 4.9. Druhý, funkční blok, obsahuje příkaz *"global.set("epsw1",msg.payload);"*, který zajišťuje, aby se vygenerované číslo uložilo do globální proměnné. Hodnota této proměnné je využívána dále při kontrole hodnoty zadaného autorizačního kódu.



Obr. 4.9: Zaslání autorizač. e-mailu

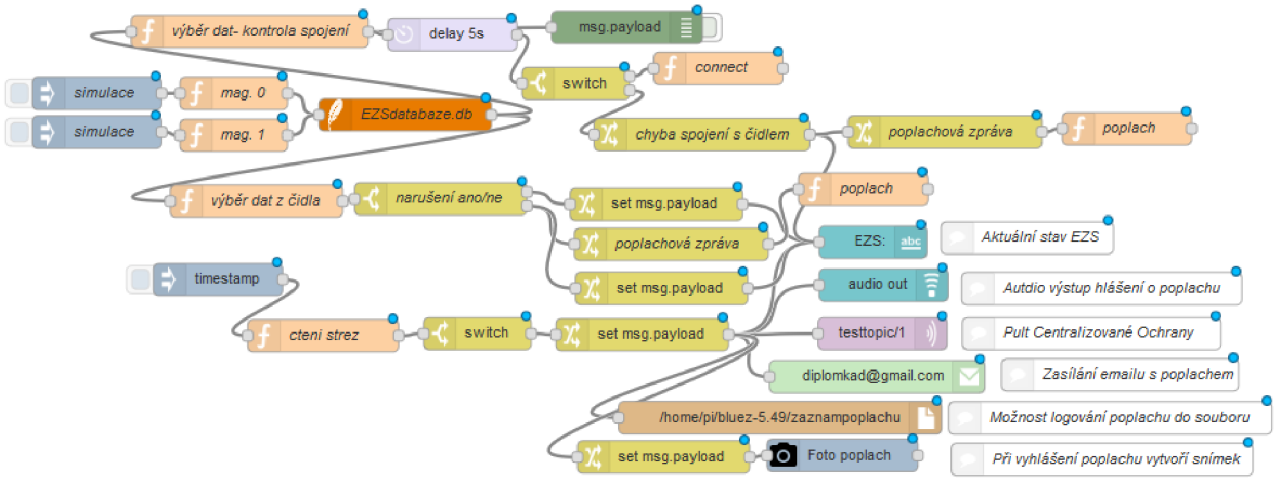
Dále na obrázku 4.6 vidíme bloky typu Switch, Delay a další, jejichž vysvětlení nalezneme v kapitole 4.1, případně na oficiálních stránkách Node-Redu, které byly zmíněny výše.

Na obrázku 4.10 nalezneme k programové části změně hesla odpovídající část vizuálního uživatelského prostředí, které jsou navzájem provázány. Node-Red umožňuje hodně možností, jak vizuální prostředí programu vytvořit. I zde lze doinstalovat různé prvky jako jsou symboly, které lze umístit například na tlačítka. Další možností je doinstalování Freeboardu, který umožňuje podrobnější nastavení a lepší editaci při vytváření vizuálního prostředí.



Obr. 4.10: Změna hesla - uživatelské prostředí

Na následujícím obrázku 4.11 nalezneme ukázku vyhodnocování hodnot přijatých z čidla. Základ tvoří databáze SQLite, z které jsou vyčítány data. Následující blok switch rozhoduje zda došlo k narušení střeženého objektu nebo ne. V případě, že ne, zasílá zprávu "OK"do textboxu zobrazující stav EZS. V opačném případě případě ukládá do globální proměnné hodnotu značící poplach a následně ověřuje zda je objekt v režimu střežení. V případě, že ano, vypíše do textboxu stavu EZS varování o narušení a předá informace další části programu, které při nezadání hesla do 30 sekund vyhlásí poplach. Poplach je odeslán pomocí MQTT na broker, pomocí kterého může být připojen na pult centrální ochrany. Dále je hláška s poplachem odeslána na nastavený email, uložena do logovacího souboru a v případě připojení kamery, systém vytvoří a uloží fotografii do paměti.

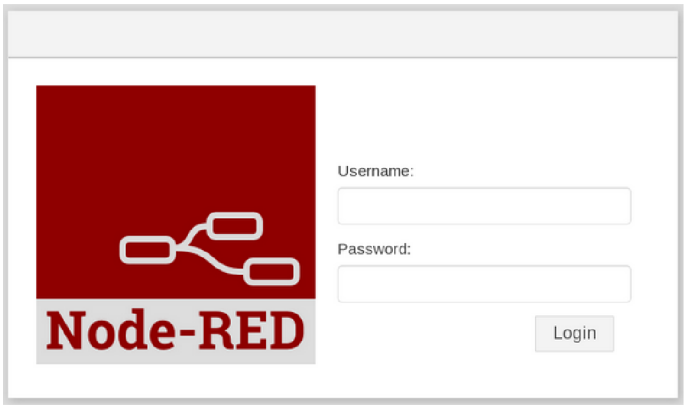


Obr. 4.11: Ukázka Node-Red čidlo

Do databáze jsou data z čidel ukládána pomocí skripu psaného v Pythonu. Pro odzkoušení bylo nutné vytvoření simulace hodnot z čidla, proto můžeme vidět dva funkční bloky, které mají výstupy směrovány do databáze. Na základě stisku timestampu, funkční bloky uloží určitou hodnotu do příslušných částí databáze.

4.4 Zabezpečení

Jelikož se jedná o elektronický zabezpečovací systém, bylo zapotřebí Node-Red zabezpečit. Proto byly vytvořeny účty pro přístup a správu Node-Redu. Samotné ovládání EZS lze provádět i přes internet. Při pokusu o přihlášení do systému, požaduje Node-Red ověření, které můžeme vidět na obrázku 4.12



Obr. 4.12: Node-Red zabezpečení

Pro vytvoření uživatelských účtů bylo zapotřebí nainstalovat balíček řešící uživatelské zabezpečení. Před samotnou instalací bylo potřebné udělat upgrade a update aby nedocházelo k problémům v rámci samotné instalace. Použité příkazy nalezneme v následujícím výpisu.

Výpis 4.2: Instalace uživatelského zabezpečení

```
1 sudo apt-get update
2 sudo apt-get upgrade
3 sudo npm install -g node-red-admin
```

Pro větší bezpečnost byla hesla vytvořena pomocí hašovací funkce. Tento proces můžeme vidět na následujícím obrázku 4.13, kde byl pomocí příkazu v terminálu vytvořen hash zadaného hesla. Jedná se o kryptografickou hashovací funkci, která zajistí, že pro ověření správně zadaného hesla stačí samotný hash a nemusí být nikde uloženo heslo. Ze samotného hashe je téměř nemožné získat původní podobu hesla. Záleží na úrovni použité zabezpečovací funkce.

```
pi@raspberrypi:~ $ sudo node-red-admin hash-pw
Password:
$2a$08$UttcxLt/Z/eGqlhuYuScE.qvp/ZJoIEn9zubUS6r78fOZkDdUWpQW
pi@raspberrypi:~ $
```

Obr. 4.13: Hash

Takto vygenerovaný hash bylo zapotřebí zkopírovat a vložit do souboru settings.js, který nalezneme v adresáři Node-Redu. Je zde možnost vytvořit více uživatelů, kterým lze nastavit libovolná práva. Část souboru můžeme vidět na následujícím obrázku 4.14, kde probíhá nastavení účtu admin.

```
GNU nano 2.7.4                                File: settings.js

    //apiMaxLength: '5mb',

    // If you installed the optional node-red-dashboard you can set it's path
    // relative to httpRoot
    //ui: { path: "ui" },

    // Securing Node-RED
    // -----------------
    // To password protect the Node-RED editor and admin API, the following
    // property can be used. See http://nodered.org/docs/security.html for details.
    adminAuth: {
        type: "credentials",
        users: [{
            username: "admin",
            password: "zZWtXTja0fBlpzD4sHCMyOCMYz2Z6dNbM6t18sJogENOMcxWV9DN.",
            permissions: "*"
        }]
    },
```

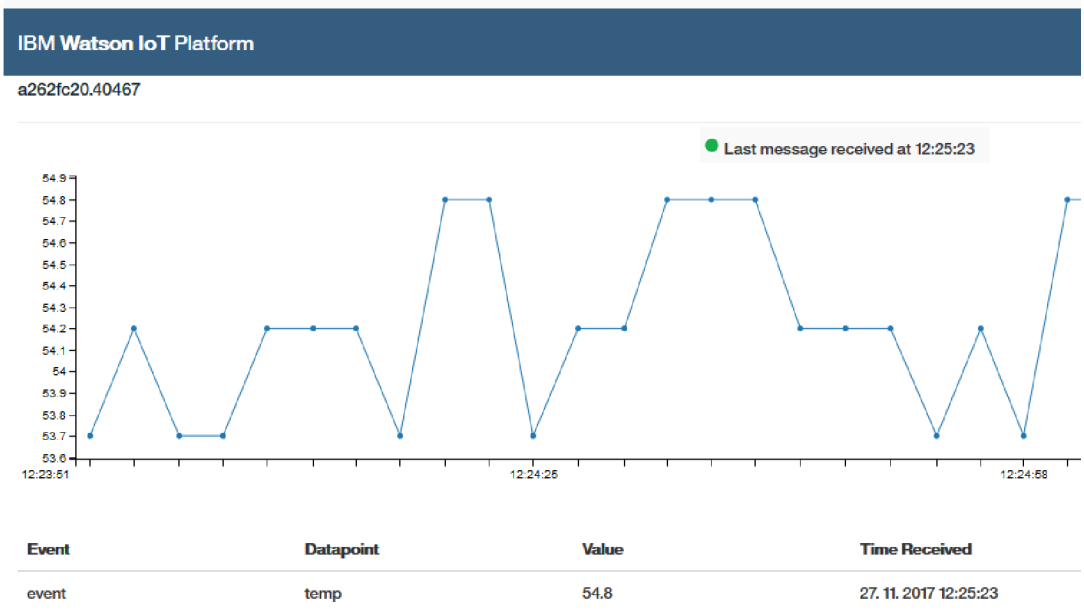
Obr. 4.14: Nastavení přihlašovacího účtu

Node-Red umožňuje více možností zabezpečení. Například vypršení platnosti ověření kódu, vlastní nastavení ověření uživatele a mnohé další, které nejsou hlavní náplní této práce.

4.5 Bluemix-IBM

V rámci této práce bylo vyzkoušeno více možností, jak odesílat data do sítě. Jednou z možností bylo odesílání dat na cloudech IBM. Přesněji řečeno komunikace pomocí platformy Bluemix.

Tato platforma IBM slouží k vytváření, nasazení či správě cloudových aplikací. V rámci odzkoušení IoT byl založen účet na Bluemixu pouze ve free rozsahu. Tedy s omezením určitých funkcí. Pomocí Node-Redu, jehož server běží přímo na Raspberry, bylo naprogramováno zasílání dat do cloudu IBM. Tato data byla uložena a následně zobrazena v přehledných grafech, které bylo možné sledovat v reálném čase. Jako zkušební data pro přenos byly využity teplotní hodnoty z čidla na CPU Raspberry Tyto hodnoty byly snímány každé 3 sekundy a následně odesílány. Část přijatých dat můžeme vidět na obrázku 4.15, který zachycuje určitý časový úsek teploty CPU.

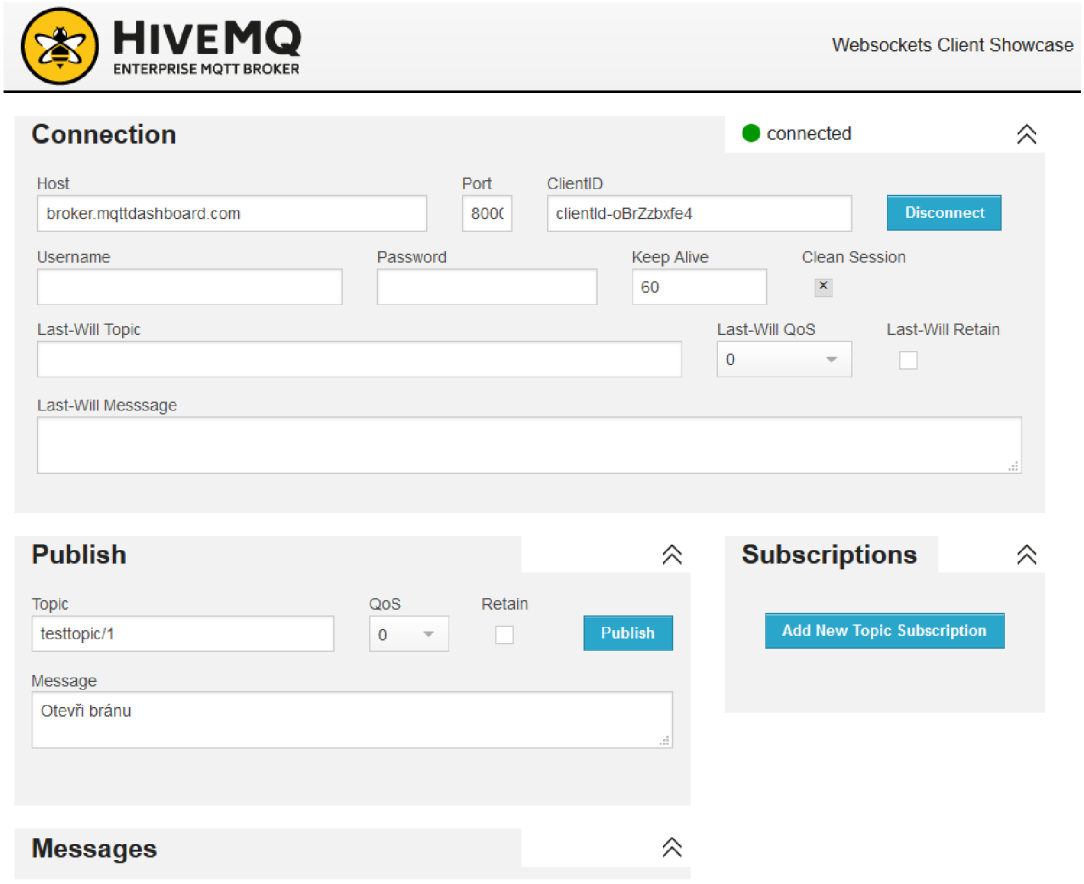


Obr. 4.15: IoT-IBM Bluemix

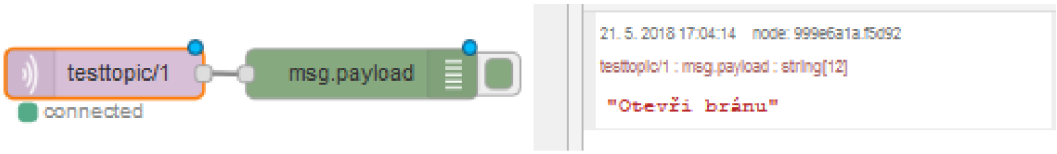
4.6 MQTT - HiveMQ

HiveMQ je společnost zabývající se MQTT službami. Jejich broker umožňuje správu velkého množství zařízení komunikujících pomocí MQTT. Zdarma umožňují otestovat Publish i Subscription. Tedy vysílání i přijímání MQTT dat. Internetové stránky zpřístupňující tyto možnosti nalezneme na následujícím odkazu: `http://www.hivemq.com/demos/websocket-client/`. Na následujícím obrázku nalezneme testu propojení Node-Redu a daného brokeru. 4.16 Můžeme si všimnout, že zde byla odeslána zpráva "Otevři bránu"a její doručení v Node-Redu vidíme na obrázku 4.17, kde je zpráva zobrazena v konzoly. Zasílání dat z Node-Redu bylo také ověřeno.

Pro nastavení komunikaci je zapotřebí znát adresu serveru, komunikační port, klientské číslo a hodnotu topicu, která je takovým identifikátorem dané komunikace. V případě zabezpečení komunikace, je zapotřebí znát i uživatelské jméno s heslem.



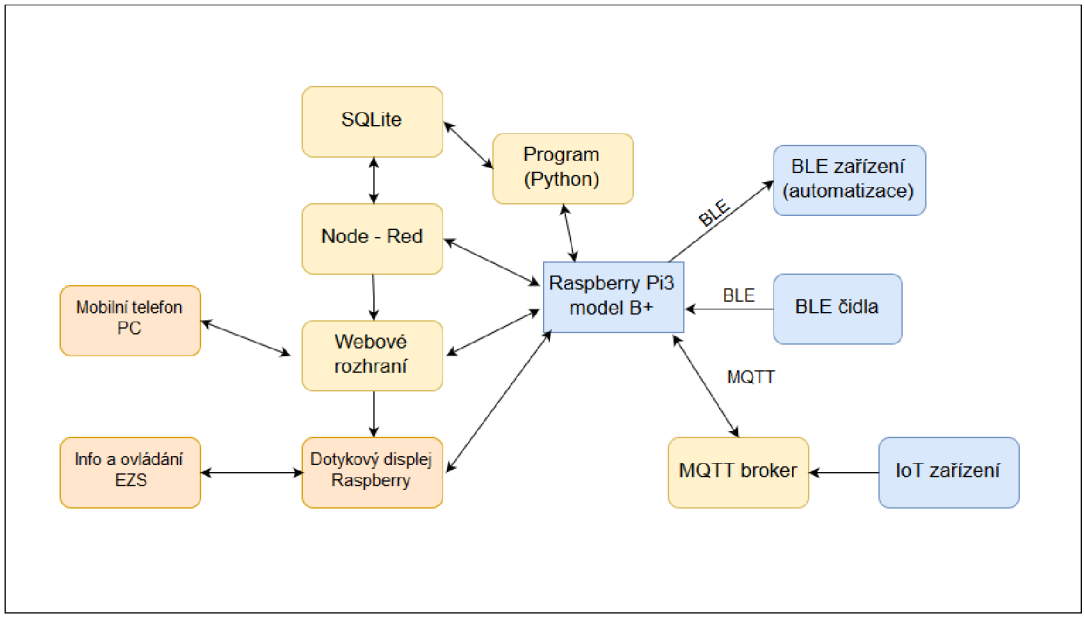
Obr. 4.16: Ukázka HiveMQ



Obr. 4.17: Node-Red příjem MQTT

5 ELEKTRONICKÝ ZABEZPEČOVACÍ SYSTÉM

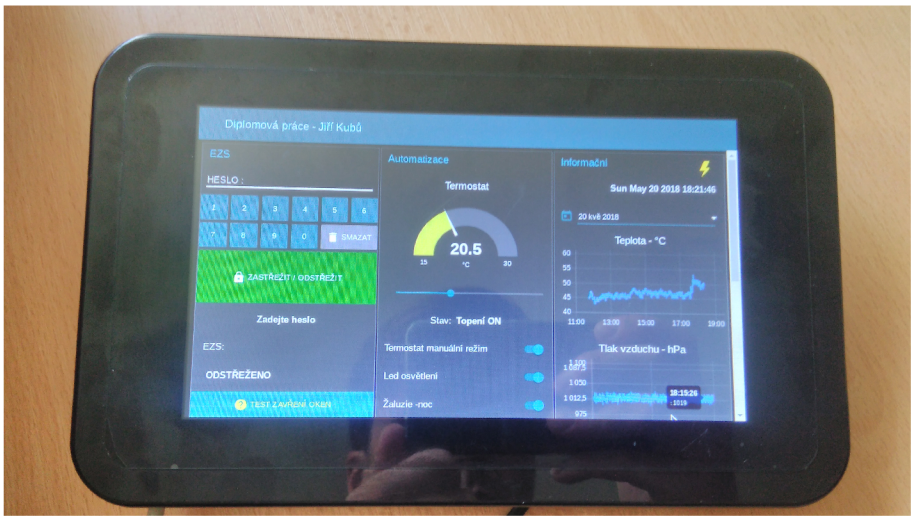
Hlavním cílem této práce byla realizace vlastního návrhu elektronického zabezpečovacího systému s možností rozšíření o domácí automatizaci. V rámci diplomové práce byl EZS sytém realizován, jehož aktuální schéma nalezneme na následujícím obrázku 5.1.



Obr. 5.1: Funkční schéma EZS

5.1 Ústředna EZS systému

Základním prvkem celého EZS systému je Raspberyy Pi3 B+, které tvoří ústřednu. Hlavním ovládacím prvkem je 7"dotykový displej připojen přímo na RPi. Tento displej je společně s Raspberry Pi umístěn ve vnitřní části domu poblíž hlavního vstupu, tak aby bylo možné rychle daný objekt odstřežit. Předpokládejme, že vytvářená EZS by měla zálohované napájení například pomocí akumulátorů, stejně jako domácí router, který zajišťuje propojení domácí sítě s Internetem. Ústředna je do domácí sítě připojena pomocí WiFi, případně Ethernetu. Tato práce je zaměřena na bezdrátovou komunikaci pomocí bluetooth low energy, proto jsou zde řešeny především bezdrátové senzory založené na této technologii. Ústředna je nastavena tak, aby se ihned po připojení napájení automaticky spustily veškeré potřebné programy a přes celou dotykovou obrazovku bylo zobrazeno ovládací grafické rozhraní. Náhled na realizovanou ústřednu nalezneme na následujícím obrázku 5.2. Samotné RPi je uložené v krytu pod dotykovým displejem.



Obr. 5.2: Grafické rozhraní

Další možností ovládání EZS je pomocí webového rozhraní. Jak již bylo řečeno, Node-Red zobrazuje grafické rozhraní v lokální síti na portu 1880. Zpřístupnění grafického ovládání z Internetu lze zajistit využitím veřejné IP adresy. Výhodou tohoto systému je využití Node-Redu, ve kterém lze jednoduše pracovat s MQTT. Proto rozšíření o další IoT prvky není až tak náročné. Ovládání přes internet by se tedy nemuselo řešit zřizováním veřejné IP, ale zapracováním ovládání určitých funkcí pomocí MQTT. Uživatelská ovládání by probíhalo například pomocí androidové aplikace v telefonu založené na MQTT.

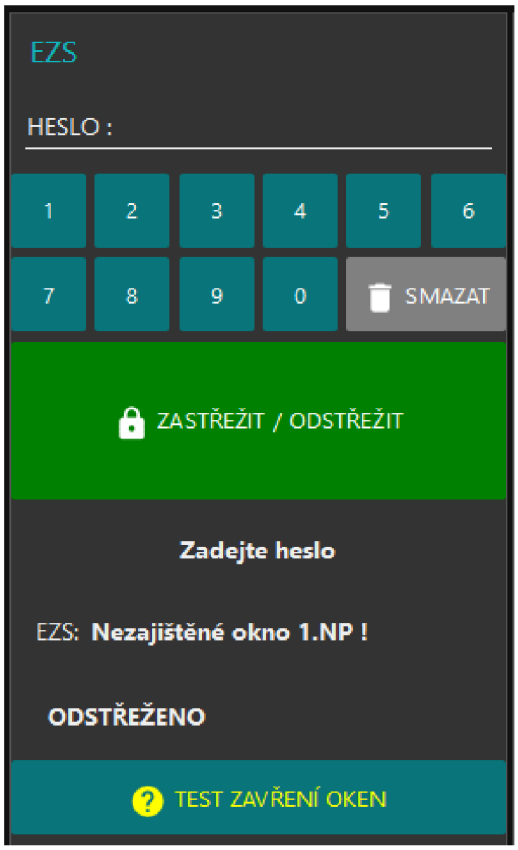
Celý systém je rozdělen na 3 části, které mají různé funkce. V následujících odstavcích jsou popsány jejich jednotlivé funkce a možnosti, které nabízejí.

5.1.1 EZS část

Bezesporu se jedná o nejdůležitější část EZS, jejíž grafické prostředí vidíme na obrázcích 5.3 a 5.4. První oddíl je věnován samotné obsluze zabezpečovacího systému. V horní části nalezneme textbox na zadání hesla. Kvůli přizpůsobení všem zařízením, které se budou k danému systému připojovat, jsou zde vytvořeny klávesy pro rychlé zadání hesla. V případě mobilních zařízení se ihned po stisknutí textboxu vynoří klávesnice. Grafické prostředí bylo testováno na 7 různých zařízeních a pouze u jednoho byl zjištěn občasný problém se zobrazením klávesnice, které trvalo delší dobu. Tento problém byl odstraněn instalací jinou virtuální klávesnicí.

Tlačítkem zastřežit / odstřežit dojde po zadání správného hesla ke změně stavu. Aby bylo možné zabezpečovací systém odstřežit pomocí hlavní ústředny, vstupem do objektu dochází nejprve k narušení střežení. V daný moment je nastaven časový limit na 30 sekund a pro urychlení procesu odstřežení je zadávané heslo kontrolováno

průběžně, aby bylo co nejvíce zamezeno falešným poplachům. V případě, že bylo střežení objektu narušeno, systém vyžaduje zadání hesla do 30 sekund i v případě, kdy byl zaznamenaný okenní magnet zpátky sepnut.



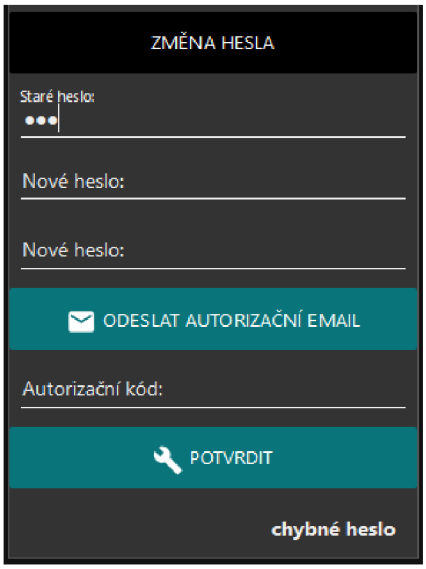
Obr. 5.3: EZS část

Jak si můžeme všimnout na obrázku 5.3, systém uvádí aktuální informace o stavu EZS. Zda-li je objekt odstřežen nebo zastřežen, zda je heslo přijato, nebo kolik je celkem času na odstřežení. V případě nižší kapacity baterií v bezdrátových čidlech se zde vypíše příslušné varování.

Další funkce tohoto systému je možnost kontroly aktuálních stavů jednotlivých adresovaných čidel. Ukázka je zde vytvořena na okruhu magnetických senzorů oken, kdy po stisku tlačítka, systém ověří stav příslušných čidel. Jelikož jsou jednotlivá čidla navržena s adresací, lze dané čidlo přesněji lokalizovat.

Následující obrázek 5.4 zobrazuje možnost změny hesla. Ve spodní části obrázku se zobrazují veškeré instrukce tak, aby bylo vše vyplněno správně a mohlo dojít ke změně hesla. V první řadě je zde ověření pomocí starého hesla. Dalším krokem je dvojité zadání nového hesla a autorizačního kódu, který je po stisku příslušného tlačítka zaslán na administrátorský email. V případě, že jsou všechny náležitosti správné, program změnu hesla potvrdí a odešle kontrolní email s informací o změně

hesla. Nastavení práv změny hesla by se v Node-Redu nechalo nastavit v závislosti na uživatelském přihlášení.



Obr. 5.4: EZS část hesla

Jako ukázka možností přidání do programu i audio výstup, systém hlásí požadavek na zadání hesla a v případě vyvolání poplachu hlásí "ALARM". V případě, že k ústředně není připojen žádný reproduktor, hlášení se objeví jako malé sdělení na obrazovce, kde sdělí, že zařízení říká "ALARM". V ostatních zařízení, jako je například mobilní telefon, jsou hlášení oznamována jazykem, který je pro dané zařízení nastavené jako výchozí.

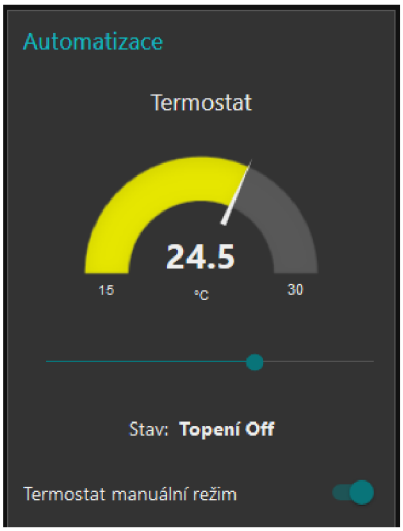
5.1.2 Automatizační část

Jako ukázka možností domácí automatizace byl vytvořen termostat, který by mohl ovládat například vytápění domu. Na následujícím obrázku 5.5 nalezneme grafické zobrazení termostatu, jehož hlavní funkcí je spínání určitého obvodu v závislosti na požadované teplotě.

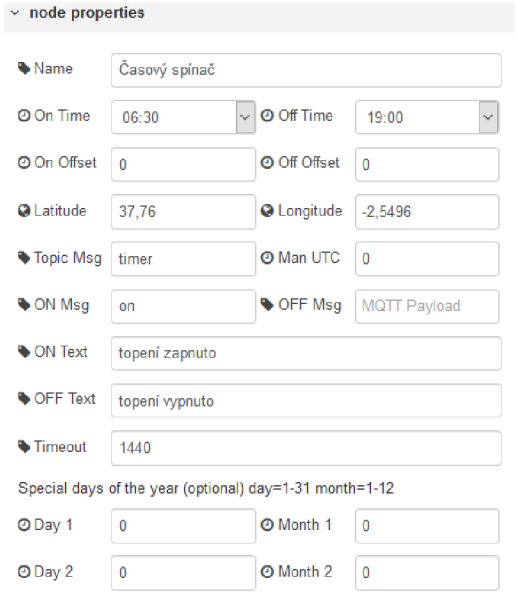
Možností při vytváření grafiky jsou zde poměrně propracované. Nalezneme zde různé efekty jako je například proměnné barvy prostředí,grafů apod. v závislosti na akci či hodnotě dané proměnné. Výběr barev není pouze ze základních řad, ale lze vybrat jakékoliv odstíny.

Vytvořený termostat umožňuje dva režimy provozu. Prvním je režim manuální, který neustále udržuje požadovanou teplotu. Druhý režim je plně automatický, který se řídí pomocí naprogramované logiky. Ta byla docílena pomocí doinstalovaného bloku BigTimer, jehož část programovacího prostředí vidíme na následujícím obrázku 5.6. Umožňuje detailní nastavení s ohledem na minuty, hodiny, dny, týdny

atd. včetně nastavení určitých vyjímek, například lišícího se od běžného týdenního režimu.

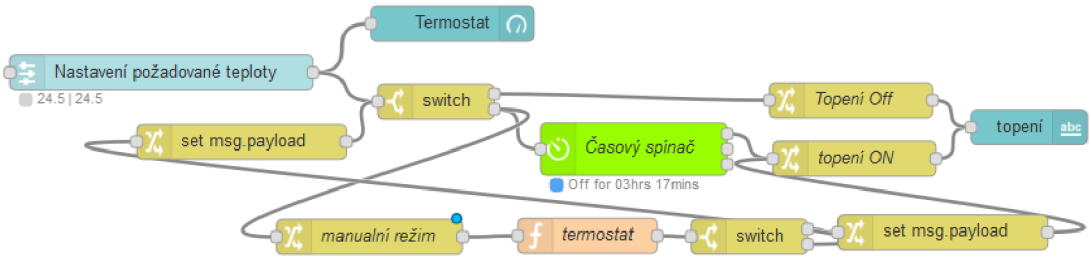


Obr. 5.5: Automatizace



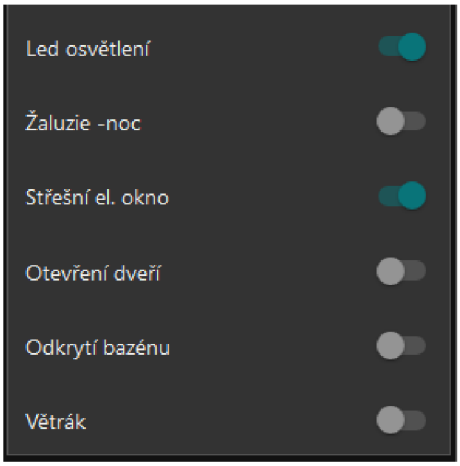
Obr. 5.6: BigTimer - část nastavení

Pro ukázku, nalezneme na následujícím obrázku 5.7 část funkčních bloků řešící sekci termostatu.



Obr. 5.7: Ukázka termostat Node-Red

Druhou částí této automatizační sekce je samotné ovládání jednotlivých zařízení. Na následujícím obrázku 5.8 vidíme ukázku 8 zařízení s možností bezdrátového ovládání z grafického prostředí systému. Bližší popis této funkce nalezneme v kapitole 5.1.5, kde se daný princip řeší v části dokumentace k programu psanému v jazyce Python.



Obr. 5.8: Automatizace

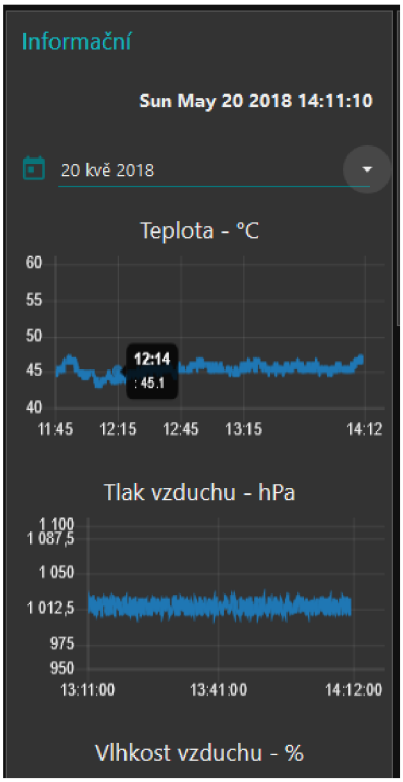
Samotná automatizace by nemusela zpracovávat pouze pokyny zadané v rámci grafického rozhraní Node-Redu, ale mohla by být ovládána přijatým příkazem z MQTT serveru. Uvedem-li příklad, využití by se našlo u otevírání příjezdové brány. Pomocí androidové aplikace v mobilu, by byl odeslán požadavek na otevření brány určitému MQTT brokeru, který by zprávu odeslal do ústředny EZS. Výsledkem by tedy bylo umožnění ovládání příjezdové brány určitým uživatelům, kteří by nemuseli mít přístup do ovládání celé EZS.

Dnešní doba je plná moderních a chytrých věcí, které lze ovládat pomocí BLE. Proto existuje nespočet zařízení, které by se daly s tímto systémem poměrně snadno

propojit. Ať už se jedná o velký elektrospotřebič typu automatická pračka, nebo menší kuchyňský kávovar, který začne připravovat kávu sotva se blížíte k domu.

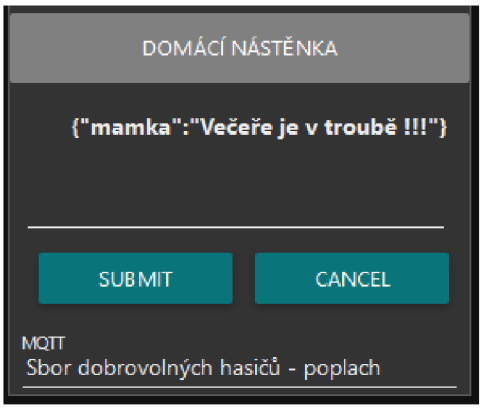
5.1.3 Informační část

Jak samotný název vypovídá, jedná se o část grafického prostředí, kerá má za úkol informovat. Jako ukázku možností vidíme na následujícím obrázku 5.9 aktuální datum, kalendář po jehož otevření se zobrazí možnosti výběru měsíce atd. Jelikož většinou každá domácnost vlastní meteostanici, i zde se můžeme podívat na příjem dat z čidel, jejichž hodnoty jsou v rámci práce nasimulované. Jedná se celkemm o tři grafy zaznamenávající teplotu, tlak a vlhkost vzduchu.



Obr. 5.9: Informační část

Dále do této části byla zahrnuta práce s MQTT, kdy ústředna přijímá data z MQTT brokeru, které jsou následně zobrazena ve spodní části následujícího obrázku 5.10. V rámci ukázky byla zpráva "Sbor dobrovolných hasičů - poplach"odeslána na MQTT broker, ze kterého byla zpráva ihned zobrazena v grafickém prostředí systému a dále zaslána na audio výstup. Dále zde můžeme vidět vytvořenou takzvanou domácí nástěnku, která slouží například pro zanechání krátkých vzkazů.

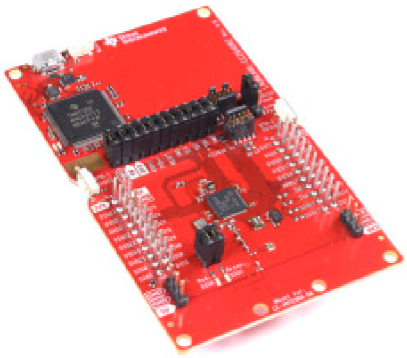


Obr. 5.10: Informační část2

5.1.4 EZS čidla

Další důležitou kapitolou jsou EZS čidla. V této práci je řešena komunikace pomocí BLE, ale nabízí se zde možnost i komunikace s kabelovými čidly. Například pomocí integrované sběrnice I2C.

V rámci této práce byl navržen koncept celé komunikace, která měla být v původním plánu propojena s bakalářskou prací zaměřenou na výrobu BLE čidla. Tato spolupráce se vyvíjela poměrně dobře, kdy byly druhé straně předloženy návrhy komunikace a celkové požadavky na koncept systému. Byla odzkoušena základní komunikace a probrána daná problematika. Bohužel zapracování základních domluvených požadavků nebylo včasně vyřešeno, a proto bylo finální testování prováděno pouze v rámci této diplomové práce. Pro testování byl k dispozici stejný microcontroller, jako měla zmíněná bakalářská práce. Jedná se o vývojový kit LaunchPad, který si můžeme prohlédnout na obrázku 5.11.

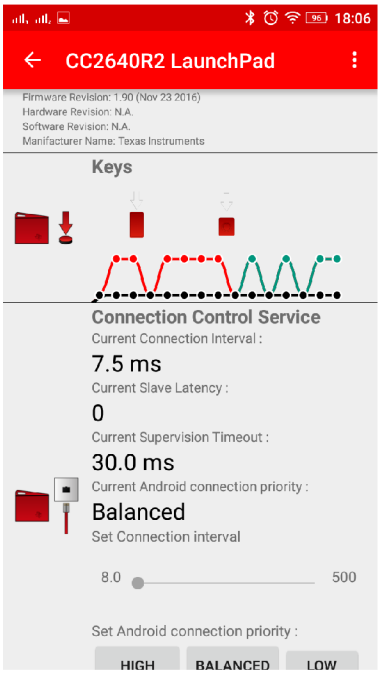


Obr. 5.11: LaunchPad CC2640R2

V této práci LaunchPad sloužil především jako zdroj dat vysílaných pomocí BLE, případně jako cíl pro ukládání dat využívané k domácí automatizaci. Tento

vývojový kit pochází od společnosti Texas Instruments a přesné označení modelu je CC2640R2. Tento microcontroller je určen především pro IoT aplikace, jelikož jeho hlavní funkcí je komunikace pomocí Bluetooth low energy.

Tato verze microcontrolleru má implementované dvě tlačítka po jejichž stisku se pomocí BLE odešlou informace o dané akci. Texas Instruments poskytuje na GoogglePlay zdarma aplikace pro správu zařízení tohoto typu. Pro tuto práci aplikace posloužila k prvotnímu ověření, zda jsou data reálně vysílána. Při hledání již zmíněného problému s komunikací bylo pomocí této aplikace ověřováno, zda závada není způsobena špatným čidlem. Prostředí z mobilního telefonu při testování můžeme vidět na obrázku 5.12.



Obr. 5.12: Příjem dat Android aplikace

5.1.5 Python skript

Programovací stránka práce byla rozvržena následujícím způsobem. První, větší část, byla vytvářena v Node-Redu, která zajišťuje především samotnou logiku systému a vyhodnocování dat. Druhou, programovací částí bylo zapotřebí vyřešit zajištění konektivity zařízení, přenos dat, jejich zpracování a následné uložení do databáze, ze které následně čerpal Node-Red. Tato část byla řešena v programovacím jazyku Python. Vytvořený skript, který nalezneme v příloze na CD, byl na RPi nastaven běžným linuxovým postupem na automatické spuštění po startu systému.

Skript byl vytvořen pro testování s BLE zařízením LaunchPad. Samotná komunikace by měla být totožná s jinými druhy čidel pomocí BLE . Hlavní odlišnost by

mohla nastat při ověřování přístupového kódu, kterému by se skript musel přizpůsobit.

V první řadě bylo zapotřebí nainstalovat na RaspberryPi Python. Některé verze Raspbianu obsahují základní, či starší verze, které je možné aktualizovat, ale mnohem jistější postu je smazání starších verzí a nainstalování aktuální. Příkaz nalezneme v následujícím výpisu.

Výpis 5.1: Instalace Python

```
1    sudo apt-get update
2    sudo apt-get upgrade
3    sudo apt-get install git build-essential python-dev
4         python-pip
```

Dalším potřebným krokem je instalace správných knihoven pro práci s Bluetooh. V našem případě se jedná o instalaci balíčku BlueZ. Jak se při prvních pokusech ukázalo, zde je vysloveně nutností předem odinstalovat starší verzi BlueZu, jelikož následně způsobovala problémy. Proto bylo aplikováno odinstalování a následné prohledání adresářů, kde bylo zapotřebí ručně odstranit veškeré pozůstatky.

Na následujících webových stránkách http://www.bluez.org/download/ nalezneme vždy aktuální verzi balíčku kterou zde můžeme i zdarma stáhnout. V našem případě se jednalo o verzi Bluez-5.49.

Následující instalace probíhala pomocí příkazů uvedených v následujícím výpisu. Důležitá byla správná instalace bez chyb či závažnějších varování, které by mohly samotnou práci s bluetooth komplikovat.

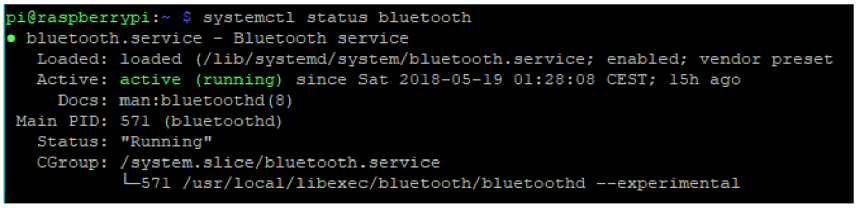
Výpis 5.2: Instalace Bluez

```
1    wget http://www.kernel.org/pub/linux/bluetooth/bluez-5.49.
2                  tar.xz
3    tar xvf bluez-5.49.tar.xz
4    cd bluez-5.49
5    ~/bluez-5.49 $ sudo apt-get updatesudo apt-get install -y
6                  libusb-dev libdbus-1-dev libglib2.0-dev
7                  libudev-dev libical-dev libreadline-dev
8    ~/bluez-5.49 $ sudo apt-get update
9    ~/bluez-5.49 $ sudo apt-get install -y libusb-dev libdbus-
10                 1-dev libglib2.0-dev libudev-dev libical-dev
11                 libreadline-dev
12   ~/bluez-5.49 $ ./configure
13   ~/bluez-5.49 $ make
14   ~/bluez-5.49 $ sudo make install
15   ~/bluez-5.49 $ sudo systemctl daemon-reload
```

Pokud celá instalace proběhla úspěšne, můžeme ji ověřit následujícím příkazem, který nám podá informace o stavu bluetooth. Pokud není v aktivním režimu, použijeme příkaz na druhém řádku následujícího výpisu. Obrázek 5.13 zobrazuje správný status bluetooth po instalaci nové verze BlueZ.

Výpis 5.3: Kontrola Bluez

```
1    ~/bluez-5.49 $ systemctl status bluetooth
2    ~/bluez-5.49 $ sudo systemctl start bluetooth
```



Obr. 5.13: Status bluetooth

V rámci práce byly využívány dvě čidla, přesněji řečeno dva LauchPady, které měly odlišný firmware. Hlavním rozdílem byla jiná struktura UUID, kdy bylo zapotřebí dohledat správné uživatelské UUID, které by mohlo být využíváno pro danou komunikaci. Nejdříve bylo zapotřebí zjistit identifikační údaje o čidlech, které bylo prováděno pomocí Node-Redu. Další kroky byly prováděny pomocí terminálu. Pomocí příkazů v následujícím výpisu byly zjištěny potřebné informace o hlavních UUID a následně jejich detaily.

Výpis 5.4: Získávání informací UUID

```
~ $ sudo hciconfig hci0 up
~ $ sudo hcitool lescan
LE Scan ...
1C:B5:21:A2:C0:C0 (unknown)
B0:91:22:69:FB:C6 Project Zero R2
~ $  gatttool -I -b B0:91:22:69:FB:C6
[B0:91:22:69:FB:C6][LE]> connect
Attempting to connect to B0:91:22:69:FB:C6
Connection successful
[B0:91:22:69:FB:C6][LE]> primary
attr handle: 0x0001, end grp handle: 0x0007 uuid:
                    00001800-0000-1000-8000-00805f
attr handle: 0x0008, end grp handle: 0x0008 uuid:
                    00001801-0000-1000-8000-00805f
attr handle: 0x001c, end grp handle: 0x0020 uuid:
                    f0001110-0451-4000-b000-000000
attr handle: 0x0009, end grp handle: 0x001b uuid:
                    0000180a-0000-1000-8000-00805f
```

Detailnější informace při připojení na čidlo pomocí příkazu "characteristics", jehož malou část můžeme vidět v následujícím výpisu. Na posledních řádcích nalezneme ukázku vyčítání dat z určitého UUID. Data jsou v hexadecimální podobě a jejich celková velikost je určena pravidly programu daného microcontrolleru. Každé UUID má přednastavené práva přístupu. Některé lze přepisovat, jiné například pouze číst. U každého BLE čidla bylo zapotřebí uživatelské UUID, které by splňovalo nároky na přenos dat z čidla a zároveň, aby nebylo využíváno i jiným programem (procesem) než námi vytvořeným.

Výpis 5.5: Výpis a čtení UUID

```
[B0:91:22:69:FB:C6][LE]> characteristics
  handle: 0x0002, char properties: 0x02, char value handle:
        0x0003, uuid: 00002a00-0000-1000-8000-00805f9b34fb
  handle: 0x0004, char properties: 0x02, char value handle:
        0x0005, uuid: 00002a01-0000-1000-8000-00805f9b34fb
  handle: 0x0006, char properties: 0x02, char value handle:
        0x0007, uuid: 00002a04-0000-1000-8000-00805f9b34fb
[B0:91:22:69:FB:C6][LE]> char-read-uuid  00002a00-0000-1000
                                        -8000-00805f9b34fb
        value: 50 00 a0 00 00 00 e8 03
```

Nyní se podíváme na hlavní části vytvořeného programu. Tento skript, psaný v Pythonu, nalezneme v příloze na CD. Jelikož se nejedná o odladěnou univerzální ústřednu, ale spíše o prototyp prezentování návrhu, pro připojení dalších BLE čidel by bylo nutné drobných úprav v daném skriptu.

Na následujícím výpisu nalezneme definování zařízení (čidel), které probíhá na základě jejich fyzické adresy. Dalším krokem je základní nastavení bluetooth adaptéru, které se provede pouze jednou a to po startu ústředny. Jedná se o reset adaptéru jeho opětovné zaregistrování agenta. Jednotlivé kroky jsou proloženy časovým prostorem pro vykonání příkazu.

Výpis 5.6: Nastavení BLE adaptéru

```
1  DEVICE = "54:6C:0E:9B:5C:2E"
2  DEVICE2 = "B8:27:EB:25:8F:43"
3
4  #  Zapinani BLE
5      zapinaniBLE = pexpect.spawn("bluetoothctl")
6      time.sleep(4)
7
8      zapinaniBLE.sendline("power␣off")
9      time.sleep(4)
10
11     zapinaniBLE.sendline("power␣on")
12     time.sleep(4)
13
14     zapinaniBLE.sendline("agent␣on")
15     time.sleep(4)
16     print(zapinaniBLE.before),
17
18 print("\n␣Komunikace␣Bluetooth␣Low␣Energy␣zarizeni:"),
19 print(DEVICE),
```

Dalším krokem je scanování BLE zařízení, které provádíme pomocí příkazu gatttool. Je-li požadované zařízení (čidlo) aktivní, dojde k navázání komunikace. Pokud ne, celý proces se opakuje.

Výpis 5.7: Navázání komunikace

```
# Spousteni gatttool.
print("\n Priprava pro pripojeni , scanovani ...")
child = pexpect.spawn("gatttool -I")

# Pripojeni k zarizeni.
print(" Pripojovani k:"),
print(DEVICE),
child.sendline("connect {0}".format(DEVICE))
child.expect("Connection successful", timeout=10)
print("\n Pripojeno k BLE senzoru:"),(DEVICE),
```

Na následujícím výpisu vidíme proces získávání dat z čidla a jeho následné uložení do souboru. Je zde vidět i samotná práce s vyčtenými daty. Jedná se o převod získané hodnoty určitého bajtu z hexadecimální do desítkové soustavy a její následné uložení do soouboru.

Výpis 5.8: Získávání dat z čidla

```
child.sendline("char-read-hnd 0x002a")
child.expect("Characteristic value/descriptor: ", timeout=10)
child.expect("\r\n", timeout=10)

#Zapis do souboru
file=open("vysledky_diplomka","a")
file.write(datable +"\n")
file.close()

y = []
for x in datable.split():
    y.append(x)#znak do noveho pole
    print(x)
print(y[0])

b1 = int(y[10],16)
print("prevod:"),
print(b1)
file=open("vysledky_diplomka","a")
file.write(str(b1) +"\n")
file.close()
```

V případě rozšíření EZS o větší počet čidel, bylo zapotřebí zajistit lepší přehlednost ukládání dat, proto byla vytvořena databáze pomocí SQLite. V následujícím

výpise nalezneme vytvoření databáze s několika tabulkami. První tabulka magnety obsahuje data z čidla řešící magnetické spínače například v oknech. Obsahem této tabulky jsou údaje typu identifikace dat, přijatá hodnota(data), informace o spojení a v neposlední řadě stav baterie čidla. Myšlenkou celého přenosu je princip založený na pravidelném přenosu několika bajtů z BLE čidla přímo do ústředny. Tyto bajty jsou nosičem všech potřebných informací. V ústředně jsou následně dané bajty rozděleny a uloženy ve správném formátu do příslušných pozic databáze.

Výpis 5.9: Vytvoření částí databáze

```
1 c.execute('''CREATE␣TABLE␣magnety␣(id␣integer,hodnota␣real,
2 ␣␣␣␣␣␣␣␣␣␣conect␣real,battery␣real)''')
3 c.execute('''CREATE␣TABLE␣automat␣(id␣integer,a1␣real,a2␣real,
4 ␣␣␣␣␣␣␣␣␣␣a3␣real,a4␣real,a5␣real,a6␣real,a7␣real,a8␣real)''')
```

Druhá tabulka, jejíž vytvoření vidíme v předchozím výpisu, nese název automat. Tato část databáze je určena pro domácí automatizaci. Například budeme-li mít v domácnosti automatizační prvek komunikující pomocí BLE (v našem případě LaunchPad), bude zapotřebí informovat daný prvek o požadavcích. Uživatel si pomocí grafického rozhraní zadá požadavek na rozsvícení Led osvětlení, který bude zpracován v Node-Redu a zaznamenán změnou určité hodnoty v dané databázi. Část programu v následujícím výpisu zajistí sestavení bajtu z jednotlivých bitů uložených v databázi a následné odeslání na cílové čidlo. Každý bit reprezentuje stav určitého automatizačního zařízení. V této práci je pro ukázku vytvořeno 8 zařízení, jejichž informace jsou přenášena na jedno čidlo. Samotná délka uživatelských dat je velká 1B.

Výpis 5.10: Část domácí automatizace

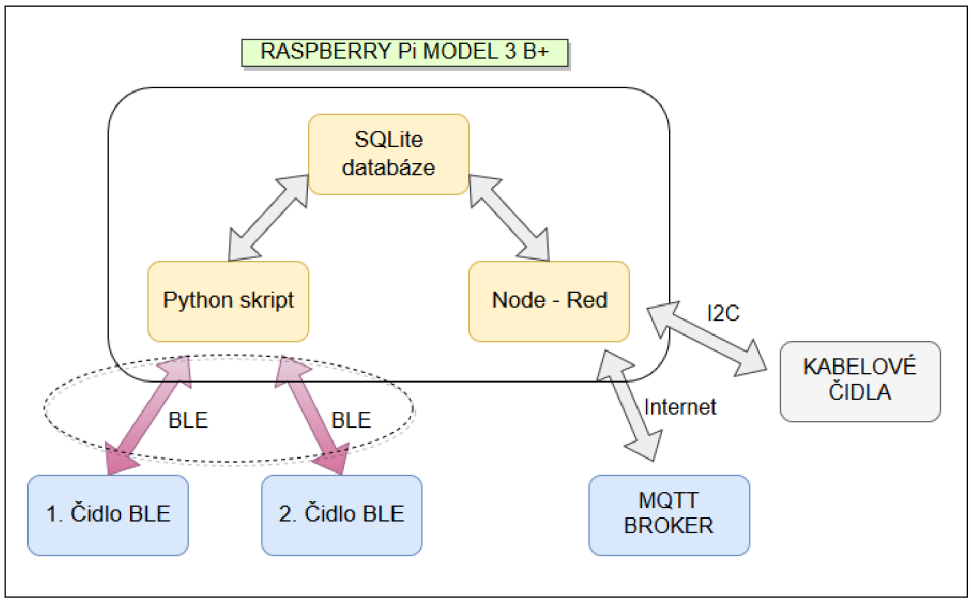
```
c.execute('''SELECT * FROM automat''')
all_rows = c.fetchall()
automat = 0
for row in all_rows:
    automat = int(row[1])<<7 | int(row[2])<<6 | int(row[3])<<5
    | int(row[4])<<4 | int(row[5])<<3 | int(row[6])<<2
    | int(row[7])<<1 | int(row[8])
print(str(automat))
string_automat = hex(automat)[2:4]+" "
print string_automat
conn.close()

#Zapis automatizace
child.sendline("char-write-req 0x0020 "+string_automat+
               "68 69 73")
child.expect("Characteristic value was written successfully ",
               timeout=10)
child.expect("\r\n", timeout=10)
```

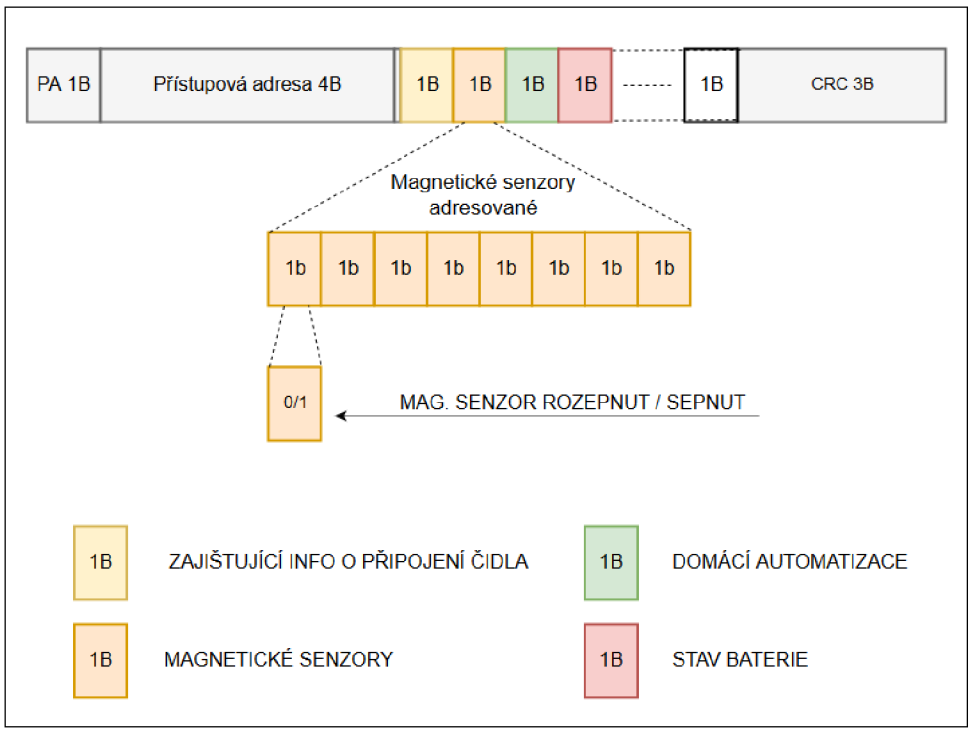
5.1.6 Princip vytvořené komunikace - shrnutí

Na následujícím obrázku 5.14 nalezneme datagram popisující samotnou komunikaci. Na Raspberry Pi model B+ je hlavním prvkem databáze SQLite, která zajišťuje propojení mezi Node-Redem a samotným skriptem, který řeší komunikaci s čidly. Node-Red program koordinuje příjem dat z MQTT brokeru a případných drátových čidel, které jsou připojeny pomocí I2C sběrnice. Dále umožňuje jako odesílací zařízení, kdy jsou naopak data odesílána MQTT brokeru, který je při případném dotazu rozešle dál. Takto lze poměrně jednoduchým způsobem propojit více oddělených systémů. Dalším výstupem Node-Redu je například e-mailová komunikace, ukládání určitých dat do takzvaného logovacího souboru a další.

Bezdrátová čidla komunikují s RaspberryPi pomocí Bluetooth Low Energy. Bylo nutné vytvořit vlastní adresovanou strukturu komunikace tak, aby byl zajištěn přenos dat s maximálním ohledem na velikost přenášených dat. Tím bylo zajištěno aby rádiový přenos byl co nejkratší a tím byla ušetřena energie v bateriích napájející čidla. Pro každé čidlo je tedy vytvořena vlastní struktura přenášených dat v závislosti na požadavcích komunikace. Ukázku řešení struktury přenášených komunikačních dat nalezneme na následujícím obrázku 5.15



Obr. 5.14: Datagram vytvořeného systému



Obr. 5.15: Vytvořená struktura dat

Vytvořený obrázek 5.15 zobrazuje ucelený princip struktury dat, které jsou nosičem informací o aktuálních stavech čidla. Jelikož je připojeno k EZS systému více bezdrátových čidel, hlavní adresace je rozlišována pomocí fyzických adres jednotlivých čidel. Komunikující čidlo má s ústřednou přesně stanovenou strukturu přenáše-

ných dat, tak aby obě strany věděly co daný přenášený bit znamená. Podíváme-li se na zmíněný obrázek, vidíme, že druhý přenášený bajt je nosičem informací o Magnetických senzorech. Samotný 1 bajt obsahuje 8 bitů, kdy každý bit reprezentuje aktuální hodnotu magnetického senzoru. 0 reprezentuje normální stav a 1 naopak narušení objektu. Tímto způsobem je docílena přesná adresace jednotlivých senzorů.

Na stejném principu je založen přenos dat pro domácí automatizaci. Jednotlivý bit reprezentuje požadavek na dané zařízení, jako je například ovládané led osvětlení podhledu. Zde je hlavním rozdílem především směr komunikace. Data jsou odesílána z ústředny k danému čidlu.

První bajt navržené komunikace lze nazvat jako servisním. Aby ústředna mohla kontrolovat, zda jsou přijatá data z čidla aktuální, je daným bajtem odesíláno číslo, které funkční program pozvolna navyšuje při každé aktualizaci dat na čidle. Číslo je omezené na velikost 1B, proto celý proces probíhá dokola v omezeném rozsahu. Toto opatření zabraňuje situaci, kdy by nesprávně pracující čidlo odesílalo neaktualizovaná data. Tato kontrola slouží především pro program v Node-Redu, aby rozpoznal, zda jsou vyčítaná data z databáze aktuální.

Posledním nezmíněným bajtem z obrázku, je stav baterie. Aby byla zajištěna včasná výměna baterie v čidle, je zapotřebí informovat ústřednu o jejím stavu. Struktura rozložení bajtu je závislá na požadavku ze strany čidla. Může se jednat například o procentuální údaj (číslo) nebo jen změna určitého bitu znamenající požadavek na výměnu baterie.

5.1.7 Možné rozšíření systému

Node-Red se v poslední době velice rozšiřuje, proto je k dostání i verze pro další operační systémy, jako je například Windows. Není už tak vázaný na linuxovou platformu, proto by daný projekt bylo možné použít i na jiných zařízeních. Největší výhodu RaspberryPi je jeho nízká cena, ale zároveň vysoká kvalita s poměrně zajímavým výkonem. Tento mikropočítač se stal velmi populárním, a proto je i jeho podpora velmi velká.

Daný projekt by se nechal rozšířit například o hlasové ovládání celého systému. Tento způsob ovládání je v dnešní době velice moderní. V rámci této práce byl implementován pouze audio výstup EZS systému. Hlasové ovládání by zde bylo docela dobře řešitelné, ale jako hlavní nevýhodu vidím v tom, že kompletní zvukový záznam je odesílám na servery, kde probíhá jeho zpracování a následné odeslání výsledků zpět. Otázkou však zůstává, jak moc se daným serverům poskytující tyto funkce dá důvěřovat, aby nedocházelo k fatální ztrátě soukromí v prostředí, které by mělo být bezpečným zázemím každého z nás.

Node-Red obsahuje nespočet možností, takže rozšíření tohoto systému by bylo

limitováno spíše výkonnostní stránkou RPi. Poměrně zajímavou možností by mohlo být vytvoření multimediálního serveru pomocí RPi. Tento využívaný model B+ obsahuje dokonce Gigabitový Ethernet, který by se při přenosu větších objemů dat určitě uplatnil.

K danému realizovanému návrhu EZS by bylo možné připojit i drátová čidla například pomocí I2C sběrnice. Samotná implementace těchto čidel by byla v prostředí Node-Redu poměrně snadnou záležitostí.

Aby nebyl vytvořený systém odkázaný na komunikaci pomocí internetu, bylo by vhodné zapracovat k ústředně například GSM modul tak, aby byly například poplachové hlášení doručeny pomocí SMS zprávy.

Rozšíření systému směrem k ochraně lidského zdraví by bylo možné například propojením chytrých hodinek a samotné ústředny. Dnešní chytré hodinky využívají právě BLE komunikaci pro spojení s mobilním telefonem a mezi jejich funkce patří mnohdy například měření tlaku či tepu. Starší člověk se zdravotními problémy by byl snímán pomocí chytrých hodinek a v případě definovaných stavů by systém automaticky upozornil například rodinné příslušníky, kteří by s danou informací dále naložili.

6 ZÁVĚR

Diplomová práce byla zaměřena na IoT technologie, které jsou v dnešní době velice rozšířené. V rámci této práce byl zrealizován vlastní návrh elektronického zabezpečovacího systému pomocí RaspberryPi3 B+, které tvořilo společně s dotykovým displejem ústřednu systému. Projekt byl navržen s ohledem na možné rozšíření o domácí automatizaci, proto je hlavní část vytvořena v programu Node-Red. Při vytváření vlastní struktury komunikace byl kladen důraz na adresovatelnost jednotlivých senzorů i v případě, že jedno bluetooth čidlo obsahuje více senzorů. Samotná obousměrná komunikace mezi ústřednou a čidlem probíhala pomocí rádiového spojení. Přesněji řečeno, jednalo se o Bluetooth Low Energy verze 4.2.

Ústředna byla nakonfigurována tak, aby se po zapnutí RaspberryPi automaticky spustila v grafickém režimu pro ovládání elektronického zabezpečovacího systému. Dalším možným způsobem ovládání systému je přístup pomocí webového rozhraní po Internetu. V rámci domácí sítě je systém přístupný na lokální IP adrese. Pro ovládání celého systému z Internetu je zapotřebí zajistit veřejnou IP adresu. Druhým možným způsobem ovládání je propojení funkcí pomocí MQTT technologie, která zajistí doručení dat do ústředny EZS. Například z mobilního telefonu pomocí MQTT aplikace.

Celý vytvořený systém je postaven na individuálním nastavení každého čidla, tak aby docházelo k co nejmenším datovým přenosů a tím se snížila energetická náročnost čidla. V rámci testování systému byl využíván vývojový kit LaunchPad od firmy Texas Instruments, který tvořil základní část čidla. Senzorová nadstavba LaunchPadu byla zadána jako bakalářská práce jinému studentovi, tudíž bylo zapotřebí koordinovat společnou část. Samotné testování s kompletně vytvořeným čidlem se neuskutečnilo, jelikož při kompletaci této práce nebylo dané čidlo vyrobeno. Proto musel být vytvořený systém testován pomocí simulace, která byla provedena pomocí Node-Redu.

LITERATURA

[1] *VENKATESWARAN, Sreekrishnan. Essential Linux device drivers. Upper Saddle River: Prentice Hall, c2008,714 s. ISBN 978-0-132-39655-4*

[2] *Antonín Vojáček: IoT MQTT prakticky v automatizaci.* [online]. [cit. 2017-12-11]. Dostupné z: `https://automatizace.hw.cz/iot-mqtt-prakticky-v-automatizaci-1dil-uvod.html`

[3] *Z. Shelby, K. H.; Bormann, C.: The Constrained Application Protocol (CoAP).* [online]. [cit. 2017-12-11]. Dostupné z: `https://tools.ietf.org/html/rfc7252`

[4] *E. Rescorla, N. Modadugu: Datagram Transport Layer Security Version 1.2. 2012,* [online]. [cit. 2017-12-11]. Dostupné z: `https://tools.ietf.org/html/rfc6347`

[5] *HiveMQ: MQTT essentials.* [online]. [cit. 2017-12-11]. Dostupné z: `https://www.hivemq.com/blog/mqtt-essentials/`

[6] *M2M communications: a systems approach. 1st ed. Editor David Boswarthick, Omar Elloumi, Olivier Hersent. Chichester: John Wiley, 2012, xxiii, 308 s. ISBN 978-1-119-99475-6.*

[7] *John Kooker. Bluetooth, zigbee, and wibree: A comparison of wpan technologies [cit. 2017-12-11] Dostupné z: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.465.3779*

[8] *TOWNSEND, Kevin, Carles CUFÍ, Robert DAVIDSON a AKIBA. Getting Started with Bluetooth Low Energy. O'Reilly Media, 2014. ISBN 978-1-4919-4951-1*

[9] *Protokol MQTT: komunikační standard pro IoT Červen 2016,* [online]. [cit. 2017-12-10]. Dostupné z: `https://www.root.cz/clanky/protokol-mqtt-komunikacni-standard-pro-iot/`

[10] [online]. [cit. 2017-12-11]. Dostupné z: `https://www.statista.com/statistics/471264/iot-number-of-connected-devices-worldwide/`

[11] *Fette, I.; Melnikov, A.: The WebSocket Protocol. Září 2011,* [online]. [cit. 2017-12-11]. Dostupné z: `https://tools.ietf.org/html/rfc6455`

[12] *Fette, I.; Melnikov, A.: The WebSocket Protocol. Září 2011,* [online]. [cit. 2017-12-03]. Dostupné z: `https://www.safaribooksonline.com/library/view/getting-started-with/9781491900550/ch01.html`

[13] *Raspberry Pi, oficiální webové stránky* [online]. [cit. 2017-12-03]. Dostupné z: https://www.raspberrypi.org/

[14] *IoTBreaks - What is HCI* [online]. [cit. 2017-12-03]. Dostupné z: https://iotbreaks.com/understand-bluetooth-hci-commands-and-events/

[15] *Internet of Things (IoT) History* [online]. [cit. 2017-12-03]. Dostupné z: https://www.postscapes.com/internet-of-things-history/

[16] *ITU-T. Overview of the Internet of things.* [online]. [cit. 2017-12-03]. Dostupné z: http://handle.itu.int/11.1002/1000/11559

[17] *Abraham, J.: Understanding Bluetooth Advertising Packets. [online].[cit. 2017-12-03]. Dostupné z: http://j2abro.blogspot.cz/2014/06/understanding-bluetooth-advertising.html*

[18] *Synology oficiální webové stránky* [online]. [cit. 2017-12-03]. Dostupné z: https://www.synology.com/cs-cz/dsm/feature/desktop_backup

SEZNAM SYMBOLŮ, VELIČIN A ZKRATEK

BLE	Bluetooth Low Energy
CoAP	Constrained Application Protocol
CPU	Centrální procesorová jednotka
DTLS	Datagram Transport Layer Security
EZS	Elektronický Zabezpečovací Systém
GFSK	Gaussian Frequency-Shift Keying
HTTPS	Hypertext Transfer Protocol Secure
H2H	Human to Human
IoT	Internet of Things
IEEE	Institute of Electrical and Electronics Engineers
IP	Internet Protocol
M2M	Machine to Machine
MQTT	Message Telemetry Transport
NAS	Network Attached Storage
OS	Operační systém
PLC	Powerline Communication
QoS	Quality of Service
RPi	Raspberry Pi
SSL	Secure Sockets Layer
SSH	protokol Secure Shell
SIG	(Bluetooth) Special Interest Group
TLS	protokol Transport Layer Security
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
WPAN	Wireless Personal Area Network
WiFi	Wireless Fidelity

SEZNAM PŘÍLOH