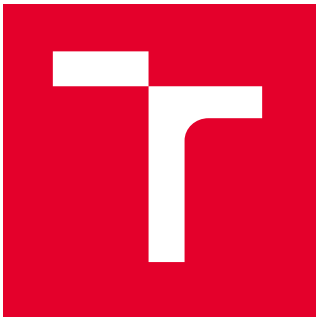


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV TELEKOMUNIKACÍ

DEPARTMENT OF TELECOMMUNICATIONS

ZAŘÍZENÍ PREZENTUJÍCÍ ZRANITELNOSTI INTERNETU VĚCI

DEVICE THAT PRESENTS THE VULNERABILITIES OF THE INTERNET OF THINGS

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — **Václav Navrátil**
AUTHOR

VEDOUCÍ PRÁCE — **doc. Ing. Zdeněk Martinásek, Ph.D.**
SUPERVISOR

BRNO 2023

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Bakalářská práce

bakalářský studijní program **Informační bezpečnost**

Ústav telekomunikací

Student: Václav Navrátil ***ID:*** 221310

Ročník: 3 ***Akademický rok:*** 2022/23

NÁZEV TÉMATU:

Zařízení prezentující zranitelnosti internetu věci

POKYNY PRO VYPRACOVÁNÍ:

Hlavním cílem práce je vytvoření komplexních kontrolních seznamů pro realizaci bezpečnostního penetračního testování zařízení internetu věci (IoT - Internet of Things). V teoretické části práce analyzujte současný stav problematiky a navrhněte kontrolní seznamy pro IoT. Kontrolní seznamy budou obsahovat popis testování problematiky v režimu White-box a Black-box. Kontrolní seznam respektuje manuální testování testerem. V rámci praktické části vytvořte hardwarové zařízení skládající se z několika IoT produktů. Zařízení bude sloužit k prezentaci zranitelností IoT a k trénování penetračního testera podle vytvořených kontrolních seznamů.

DOPORUČENÁ LITERATURA:

[1] NESHENKO, Nataliia, et al. Demystifying IoT security: An exhaustive survey on IoT vulnerabilities and a first empirical look on Internet-scale IoT exploitations. IEEE Communications Surveys & Tutorials, 2019, 21.3: 2702-2733.

[2] MENEGHELLO, Francesca, et al. IoT: Internet of threats? A survey of practical security vulnerabilities in real IoT devices. IEEE Internet of Things Journal, 2019, 6.5: 8182-8201.

Termín zadání: 6.2.2023 ***Termín odevzdání:*** 26.5.2023

Vedoucí práce: doc. Ing. Zdeněk Martinásek, Ph.D.

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

ABSTRAKT

Tato bakalářská práce je zaměřená na vytvoření kontrolních seznamů, pomocí kterých budou realizována bezpečnostní penetrační testování zařízení IoT. V teoretické části je rozebrán princip IoT, nejčastěji používané komunikační protokoly, bezpečnostní standard ISVS a z něj vyplývající kontrolní seznam. Další kapitola, tedy praktická část, obsahuje postupy popisující manuální testování několika IoT produktů. IoT zařízení byla zakoupena pro tyto účely jako jeden celek i s pomůckami na jejich testování. Celkově se provedla v praktické části čtyři komplexní testování.

KLÍČOVÁ SLOVA

Internet věcí, IoT, Zranitelnosti, Zabezpečení, Penetrační testování, White-box, Black-box, 6LoWPAN, ZigBee, Bluetooth Low Energy, BLE

ABSTRACT

This bachelor thesis focuses on creating checklists that will be used to perform security penetration testing of IoT devices. The theoretical part discusses the principle of IoT, the most commonly used communication protocols, the ISVS security standard and the resulting checklist. The next chapter, i.e., the practical part, contains procedures describing the manual testing of several IoT products. IoT devices were purchased for this purpose as a single unit along with the tools to test them. In total, four comprehensive tests were performed in the practical part.

KEYWORDS

Internet of Things, IoT, Vulnerabilities, Security, Penetration testing, White-box, Black-box, 6LoWPAN, ZigBee, Bluetooth Low Energy, BLE

NAVRÁTIL, Václav. *Zařízení prezentující zranitelnosti internetu věci*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací, 2023, 73 s. Bakalářská práce. Vedoucí práce: doc. Ing. Zdeněk Martinásek, Ph.D.

Prohlášení autora o původnosti díla

Jméno a příjmení autora:	Václav Navrátil
VUT ID autora:	221310
Typ práce:	Bakalářská práce
Akademický rok:	2022/23
Téma závěrečné práce:	Zařízení prezentující zranitelnosti internetu věci

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

Rád bych poděkoval vedoucímu bakalářské práce panu doc. Ing. Zdeňku Martináskovi, Ph.D. za konzultace, trpělivost a podnětné návrhy k práci.

Obsah

Úvod **11**

1 Teoretická část **12**
1.1 Kyberkriminalita v IoT 14
1.2 Protokoly vycházející ze standardu IEEE 802.15.4 15
1.2.1 6LoWPAN 16
1.2.2 ZigBee 16
1.3 Bluetooth Low Energy (BLE) 17
1.4 Penetrační testování 20
1.4.1 BlackBox a WhiteBox testování 20
1.4.2 IoT Security Verification Standard - ISVS 21
1.5 Kontrolní seznam 25
1.5.1 Chytrá žárovka - BLE 25
1.5.2 IoT zařízení pracující na frekvenci 433MHz 26
1.5.3 Zařízení komunikující v síti ZigBee 27
1.5.4 Chytrá zásuvka - Wi-Fi 28

2 Praktická část bakalářské práce **30**
2.1 Black Box testování chytré LED žárovky 30
2.2 BLE LED žárovka - exploitace 31
2.2.1 Použitá zařízení 31
2.2.2 Průzkum a odposlech 32
2.2.3 Převzetí kontroly nad žárovkou 35
2.2.4 Řešení případných problémů 36
2.3 Black Box testování odchycené radiové komunikace 38
2.4 Reverzace rádiových signálů 38
2.4.1 Použité Prostředky 38
2.4.2 Příprava zařízení 39
2.4.3 Identifikace frekvence s RTL-SDR 41
2.4.4 Zachycení a zpracování signálů 43
2.5 Black Box testování komunikačního prostředí ZigBee 45
2.6 ZigBee exploitace 45
2.6.1 Použité Prostředky 45
2.7 White Box testování mobilní aplikace 52
2.8 Kompromitace mobilní aplikace chytré zásuvky 53
2.8.1 Použité Prostředky 53
2.8.2 Příprava zařízení 54

2.8.3 Analýza mobilní aplikace 54
2.8.4 Odchyt síťového provozu 57
2.8.5 Analýza dat paketů 60

Závěr **62**

Literatura **63**

Seznam symbolů a zkratek **68**

A Seznam řetězců (Příloha) **71**

Seznam obrázků

1.1 Ilustrace IoT . . . 12
1.2 Graf vývoje počtu IoT a ne-IoT zařízení v čase . . . 13
1.3 Vrstvy ZigBee protokolu . . . 17
1.4 Vrstvy BLE protokolu . . . 18
1.5 Přehled oblastí požadavků ISVS . . . 21
2.1 Chytra BLE LED žárovka Govee . . . 31
2.2 Bluetooth USB adaptér . . . 31
2.3 Ubertooth One . . . 32
2.4 Zjištění adresy zařízení . . . 32
2.5 Kontrola verze firmwaru a knihovny . . . 33
2.6 Wireshark Konfigurace . . . 33
2.7 Přidání destinace . . . 33
2.8 Start odposlechu . . . 34
2.9 Filtrování odchycených paketů ve Wiresharku . . . 35
2.10 Navázání spojení pomocí nástroje Gatttool . . . 35
2.11 Změna barvy žárovky . . . 36
2.12 Ukázka přílohy A . . . 36
2.13 Arduino Nano a 433MHz vysílač . . . 38
2.14 RTL-SDR s anténou . . . 39
2.15 Stažení knihovny RC-Switch . . . 40
2.16 Program - Arduino Nano . . . 40
2.17 Zapojení Arduino Nano + 433 MHz vysílač . . . 41
2.18 Test zařízení RTL-SDR . . . 41
2.19 Konfigurace zařízení v programu GQRX . . . 42
2.20 Prostředí programu GQRX . . . 42
2.21 Identifikace frekvence vysílání . . . 43
2.22 GNURadio graf a nastavení . . . 44
2.23 Odečtení a převedení signálu do ASCII . . . 45
2.24 Xbee modul . . . 46
2.25 Programátor XBee . . . 46
2.26 Nano IO Shield 1.0 . . . 46
2.27 Arduino Nano . . . 46
2.28 APIMote . . . 46
2.29 XCTU - Vyhledání zařízení . . . 47
2.30 XCTU - parametry portu . . . 48
2.31 XCTU - prostředí konfigurace . . . 48
2.32 ArduinoIDE - program pro ZigBee přenos . . . 49

2.33 Spuštění skriptu zbstumbler 50
2.34 Odchycení komunikace a její výpis 51
2.35 Odchycení komunikace pomocí zbwireshark 52
2.36 Zásuvka Orvibo s redukcí 53
2.37 JADx - výpis třídy SecurityAes 55
2.38 JADx - vyhledání textu v kódu 55
2.39 Ghidra - výpis funkcí v knihovně 56
2.40 Ghidra - výpis knihovny 57
2.41 Ghidra - adresa pkKey 57
2.42 Ghidra - hodnota pkKey 57
2.43 Wireshark - filtr odchycené komunikace 58
2.44 Skript na dešifrování AES 59
2.45 Nový šifrovací klíč 59
2.46 Dešifrované pakety ON/OFF 60
2.47 Porovnání požadavků a odpovědí 60

Úvod

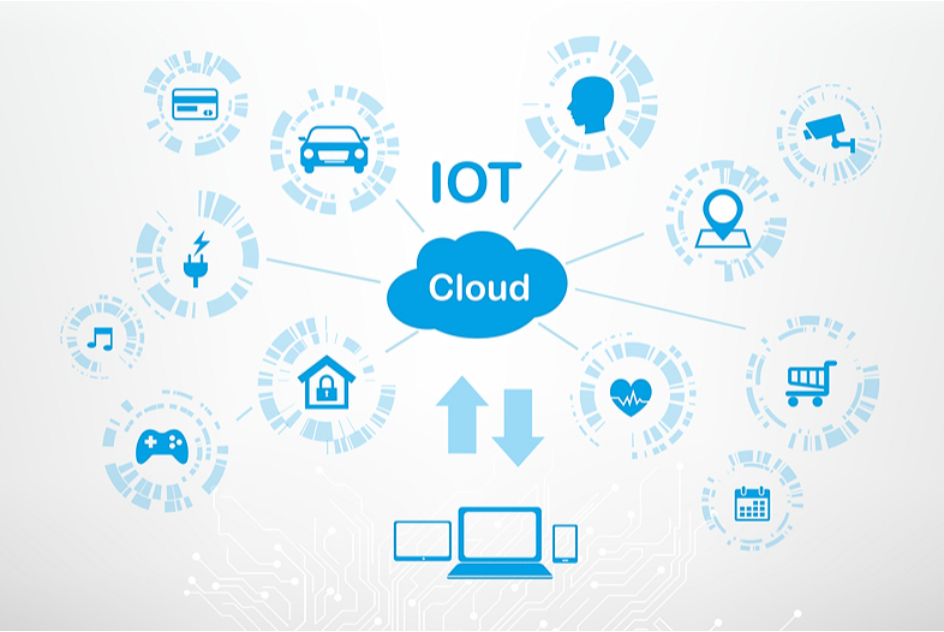
Inteligentní domácí zařízení jsou důležitou a rostoucí součástí světa internetu věcí (IoT). Jejich zavedením do našich domovů přinášíme nejen pohodlí, ale také technické problémy, na které jsme dříve nemuseli myslet.

V práci je popsán vývoj oblasti IoT a co tento pojem vlastně skrývá. Zaznamenání prvního funkčního konceptu IoT a využití ve světě. V teoretické části je také popsán současný stav a útoky za poslední období. Dále jsou rozebrány vybrané komunikační protokoly vycházející ze setandardu IEEE 802.15.4 využívané IoT zařízeními v dnešní době a popsány jejich možnosti zabezpečení.

V rámci popisu penetračního testování je rozebrán standard ISVS (IoT Security Verification Standard) a popsány jednotlivé oblasti, ve kterých udává požadavky pro kvalitní zabezpečení různých zařízení. Popis převzetí kontroly nad inteligentní žárovkou (BLE LED žárovka), průzkum, odposlech a řešení případných problémů v průběhu. Analýza radiového spektra pomocí RTL-SDR a ukázka postupu pro obnovu původních dat. Následně pak sestavení zařízení simulující komunikační prostředí ZigBee a odposlech tohoto provozu. Poslední část se věnuje analýze aplikace ovládající chytrou zásuvku pomocí reverzního inženýrství.

1 Teoretická část

Všeobecně se o Internetu věcí neboli IoT mluví jako o prostředí několika zařízení, která jsou bezdrátově či drátově propojena a jednoznačně identifikovatelná, schopna komunikovat mezi sebou a spolupracovat s dalšími věcmi či zařízeními na dosažení společných cílů. Z těchto zařízení jsou data pravidelně shromažďována, analyzována a používána například k provádění požadovaných akcí. Takovým zařízením může být dnes v podstatě cokoliv, stručný schématický popis na prvním obrázku. Můžou to být chytré telefony, meteorologické stanice, kamery, auta, domácí spotřebiče, lékařské přístroje, průmyslové stroje, dokonce lze připojit i zvířata, lidi, budovy a mnoho dalšího. Jsou schopna poskytovat bohaté informace pro plánování, řízení a rozhodování. [1]

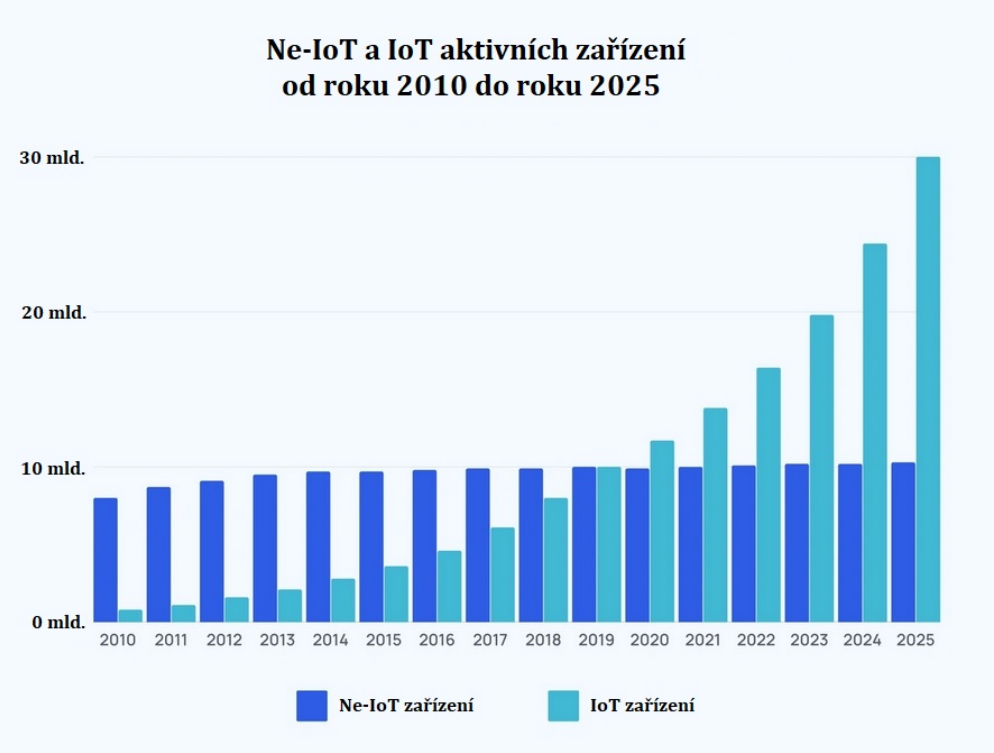


Obr. 1.1: Ilustrace IoT [2]

Prvním zařízením, které představovalo koncept IoT, byl automat na výdej pití firmy CocaCola v roce 1982. Automat byl připojen k síti ARPANET a upraven tak, aby podával informace o tom, kolik má v sobě plechovek nápoje a zda-li jsou vychlazené. V roce 1999 se Bill Joy zmínil o komunikaci dvou zařízení v jeho popisování taxonomie Internetu. V stejném roce Kevin Ashton pojmenoval systém vzájemně propojených zařízení jako "Internet of Things". [3] [4] [5]

Celosvětové výdaje na IoT se od roku 2018 meziročně zvyšují nejméně o 40 miliard dolarů. V roce 2020 dosáhnou výdaje na IoT 749 miliard dolarů. V roce 2022 se předpokládá, že celosvětové výdaje na internet věcí dosáhnou 1 bilionu

dolarů. Podle posledních dostupných údajů je k internetu věcí připojeno přibližně 13,2 miliardy zařízení. Pro rok 2025 je předpovězeno 30 miliard aktivních zařízení viz graf v obrázku 1.2. V porovnání s předpovědí z roku 2020 bylo toto číslo pro rok 2025 přehodnoceno a sníženo, jelikož na trh začaly působit dva kritické faktory a to pandemie COVID-19 a nedostatek čipů.[6] [7] [8]



Obr. 1.2: Graf vývoje počtu IoT a ne-IoT zařízení v čase [7]

Zařízení z internetu věcí jsou již snad ve všech odvětvích moderního života. Například značně zjednodušují život starým lidem a lidem s postižením, kteří díky tomu zvládají své denní rutiny bez pomoci jiných lidí, což zvětšuje jejich samostatnost a sebevědomí. Implantáty nebo nositelná zařízení poskytují lékařům informace o životních funkcích pacientů téměř v reálném čase, což zvyšuje jejich šanci na přežití. V rámci obecného bezpečí se IoT snaží minimalizovat vznik nebezpečných situací. [9]

Například vozidla připojené do sítě Internetu věcí se snaží zabránit řidiči sjet ze silnice nebo nabourat do nějaké překážky. V případě nehody umožňuje automatické nouzové volání lékařské pomoci a zaslání GPS souřadnic či nalezení nejkratší možné cesty do nemocnice. V těžebním průmyslu díky připojení těžebních strojů, buldozerů, nákladních vozidel respektive nakladačů, které se dají řídit na dálku nebo mají úplně autonomní řízení, se zaměstnanci nenacházejí v oblastech zvýšeného nebezpečí. Také v továrnách jsou IoT senzory monitorující znečištění prostředí a únik chemických

látek do vodních zdrojů. Detektory kouře, toxických plynů a teplotní senzory spojeny s poplachovým systémem předcházejí vzniku ekologických katastrof. Pomocí IoT zařízení je možné sestrojit spoustu pozoruhodných systémů, které mohou zajišťovat bezpečnost, zvyšovat efektivitu práce, dělat radost domácnostem a mnoho dalšího. [9]

Uživatelé obecně plně nerozumí tomu, jak jejich zařízení internetu věcí fungují a jaký druh informací shromažďují, ukládají a používají. Obvykle také vyjadřují "optimistickou předpojatost", kdy se nepovažují za typické oběti možných útoků. Tato předpojatost je může vést k ignorování základních doporučení týkajících se bezpečnosti a ochrany soukromí (pokud jsou vůbec k dispozici). Na rozdíl od výrobců a tvůrců zásad je koncovým uživatelům často věnována menší pozornost, v důsledku čehož nemají k dispozici žádnou oficiální a univerzální dokumentaci v podobě norem, pokynů a zásad, které by jim pomohly chránit jejich inteligentní systémy. [10]

Informační technologie umožňují uživatelům efektivní a rychlou výměnu dat, ale také nabízejí řadu výhod těm, kteří se snaží kyberprostor zneužít. Anonymita a prostorová neviditelnost na internetu znamená, že se stále více trestné činnosti přesouvá do kyberprostoru, což útočníkům umožňuje dosáhnout svých cílů rychle, snadno, a to s minimálním rizikem trestu.[11]

1.1 Kyberkriminalita v IoT

Kybernetické hrozby a kyberkriminalita zahrnují širokou škálu negativních jevů různého stupně v kyberprostoru - od špionáže, hackerských útoků, útoků DDoS až po stále častější podvody a krádeže na internetu (kompromitace internetového bankovnictví, krádeže údajů kreditních karet, falešné e-shopy, zneužití osobních údajů atd.) a další trestnou či nežádoucí činnost (šíření dětské pornografie, prodej drog a praní špinavých peněz pomocí virtuálních měn, kyberšikana, stalking, rozesílání spamu, atd.) až po projevy extremismu a zneužívání internetu k teroristickým aktivitám a propagandě (včetně např. zveřejňování návodů na výrobu výbušnin). Proto jsou hrozby v kyberprostoru jednou z hlavních výzev naší doby. [11]

Poslední závažnější incident, který se za poslední období odehrál, byl zde v Evropě. Při útoku ransomwaru na univerzitní nemocnici v Düsseldorfu zemřela v Německu žena, což je pravděpodobně první úmrtí přímo spojené s kybernetickým útokem na nemocnici. Nemocnice kvůli útoku nemohla přijímat urgentní pacienty. Zdravotnická zařízení jsou jedním z největších cílů kybernetických útoků a odborníci na kybernetickou bezpečnost již několik let varují, že většina nemocnic není připravena. Ve velké míře se spoléhají na zařízení, jako jsou radiologická zařízení, která jsou často připojena k internetu. [12]

Další podobný případ útoku se stal v Americe. Je však staršího data. Hlavními tvůrci útoku Mirai byli student Paras Jha a jeho přítel Josiah White. Poté, co v roce 2016 napsali zdrojový kód botnetu Mirai, se pomocí svého výtvoru pokusili vydírat univerzitu, kde Paras Jha studoval. [13]

Po spuštění se Mirai vymkl kontrole, protože jej kopírovali a upravovali další kyberzločinci. Od té doby se nadále projevuje v různých podobách prostřednictvím rozsáhlých útoků DDoS, které vyřadily z provozu velkou část amerického internetu. [13]

Škodlivý software Mirai nejprve skenuje IP adresy a identifikuje chytrá zařízení s verzí systému Linux známou jako Arch. Poté využívá bezpečnostní chyby v zařízeních IoT a připojuje se k síti pomocí kombinace výchozího uživatelského jména a hesla. Pokud tato nastavení nejsou změněna nebo aktualizována, může Mirai získat přístup k zařízení a infikovat je malwarem. Jakmile Mirai infikuje chytré zařízení, stane se dalším "zombie" v armádě dálkově ovládaných botů. Mirai dokonce vymaže veškerý již existující malware, aby zajistil, že zařízení bude bezpečně uzamčeno v botnetu to vše bez souhlasu nebo vědomí majitele. [13]

Zařízení IoT ovládaná tvůrcem botnetu mohou být nucena hledat další zranitelná zařízení, která by zneužila a přidávat tak nové oběti do botnetu Mirai. Vzhledem k tomu, že většina chytrých domácností nemá zavedenou komplexní ochranu sítě, zůstávají chytrá zařízení zranitelná vůči botnetu Mirai a dalším botnetům IoT. Většina zařízení napadených botnetem Mirai jsou domácí routery a kamery, ale téměř každé chytré zařízení je zranitelné vůči botnetům IoT. Stejná síťová připojení, která chytré domácnosti používají pro robotické vysavače, IP interkomy, kuchyňské spotřebiče a chytrá vozidla, jsou také potenciálními zranitelnými místy pro malware. [13]

Koncept Internetu věcí se v této době již velice rozšířil a jeho vývoj je aktivní a stále se rozvíjející. Ale stále u něj najdeme spoustu chybějících funkcionalit, mezi které patří i kvalitní zabezpečení. Úroveň zabezpečení samozřejmě s postupem let a s příchodem moderních komunikačních protokolů roste, avšak stále se nejedná o zcela zabezpečený prostor a najdeme zde spoustu zranitelností. [11]

Dále jsou rozebrány vybrané komunikační protokoly, které se nejhojněji v IoT využívají, z hlediska zabezpečení.

1.2 Protokoly vycházející ze standardu IEEE 802.15.4

Tento standard byl navržen jako základ pro skupinu komunikačních protokolů, které jsou určeny na krátkou komunikační vzdálenost, nízký datový přenos a energeticky

nenáročnou komunikaci. Poprvé byl tento standard představen v roce 2003 a definoval fyzickou vrstvu (PHY) a vrstvu, která řídí přístup k médiu, Medium Acces Control (MAC) komunikující rychlostí 250 Kbps. Poslední verze standardu byla vydána v roce 2015. [14]

Fyzická vrstva nenabízí žádné zabezpečení, ale vrstva MAC poskytuje dobré zabezpečení pomocí šifrovacího standardu AES-128 spojeného s CCM režimem (režim operací pro blokové šifry). Standard IEEE 802.15.4 uvádí tuto možnost zabezpečení jako volitelnou, ovšem většina zařízení kompatibilních s tímto standardem má od výrobce upravený hardware pro výpočet AES-128, čímž se spotřeba energie při použití tohoto zabezpečení zvýší jen minimálně. V případech, kdy se nepoužije zabezpečení na vrstvě MAC, může být zařízení stále zabezpečeno pomocí síťové nebo aplikační vrstvy. [14]

1.2.1 6LoWPAN

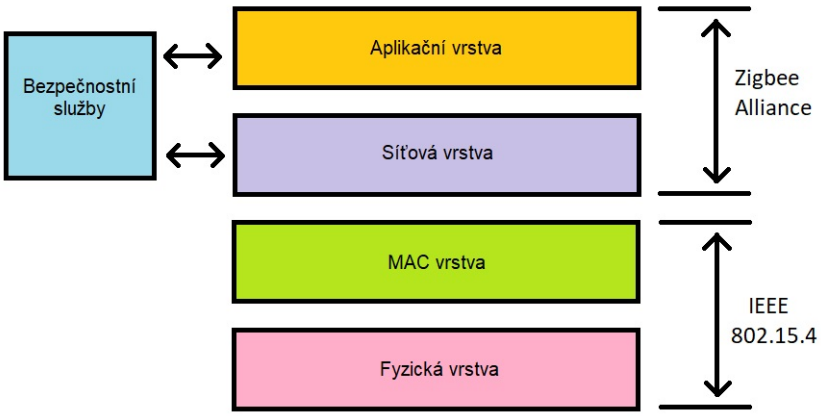
Je to nejčastěji používaný bezdrátový komunikační protokol vycházející ze standardu IEEE 802.15.4 na síťové vrstvě pro IoT zařízení, jelikož je energeticky nenáročný a založen na komunikaci pomocí IP adres. Zařízení používající tento protokol může být přímo připojeno k dalším IP sítím bez použití proxy serveru či brány (Gateway). Celý název "IPv6 over Low-Power Wireless Personal Area Networks" napovídá, že tato technologie používá pro přenos informací IPv6 pakety, které jsou upravené podle síťového standardu IEEE 802.15.4. [15] [16]

Z pohledu zabezpečení ve specifikaci 6LoWPAN není definován žádný bezpečnostní mechanismus. Spoléhá na zabezpečení s mechanismem dostupným na linkové nebo vyšších vrstvách. [14]

1.2.2 ZigBee

ZigBee protokol vyvinula ZigBee alliance. Díky němu je možné v jedné síti bezproblémově použít IoT zařízení od více výrobců. Tento komunikační protokol je rovněž založen na standardu IEEE 802.15.4, také tedy požívá PHY a MAC vrstvu viz obrázek 1.3.

Bezpečnost tohoto protokolu je založena na bezpečnostních službách poskytované standardem IEEE 802.15.4, ale nepoužívá jej přímo na MAC vrstvě, jak je to v tomto standardu definované. Používá stejný šifrovací standard AES-128 a CCM režim (Counter with Cipher block chaining Message authentication code) pro zabezpečení komunikace, jak na síťové vrstvě, tak i na aplikační. Díky tomu, že je možné sdílet jeden šifrovací klíč mezi několika vrstvami, nezvyšují se kvůli tomu tolik náklady spojené se zabezpečením. [14] [17] [18]



Obr. 1.3: Vrstvy ZigBee protokolu [18]

ZigBee má definovanou správu klíčů, což standard IEEE 802.14.5 nemá. Má definované dva typy používaných klíčů, které jsou pro běžnou datovou komunikaci. Takzvaný síťový klíč *network key*, který je sdílený mezi všemi zařízeními v síti ZigBee. Ten zajišťuje ochranu proti útočníkům nacházejících se mimo síť. A druhý linkový klíč *link key*, jenž je sdílený mezi dvěma zařízeními a ten zabezpečuje komunikaci mezi nimi samotnými. V některých případech existuje i hlavní klíč *master key*, který se používá při nastavování linkových klíčů pomocí Symmetric-Key Key Estabilishment (SKKE) protokolu, transportních *key-transport* a načítacích *key-load* klíčů používané pro zabezpečení zpráv obsahující jiné klíče. Všechny klíče používané tímto protokolem mají délku 128 bitů, ovšem nezáleží jen na jejich délce, nýbrž i na tom, jak dochází k inicializaci a instalaci těchto klíčů. Jsou dva způsoby, jak tyto klíče získat pro daná zařízení. První možnost je mít předinstalované klíče na zařízeních. A druhou je možnost získání klíče ze speciálního uzlu v síti, kterému všechna zařízení důvěřují. Tento speciální uzel se nazývá centrum důvěry *Trust Center* a také nese zodpovědnost za správu sítě. [14] [17] [18]

V případě, kdy se do sítě chce připojit nové zařízení a nemá předinstalované klíče, nastává situace, že není možné, aby Trust Center bezpečně poslalo požadované klíče tomuto zařízení. Protokol ZigBee v takovém případě dovoluje poslat nezabezpečené klíče přes síť až k nově připojenému zařízení. Tímto vzniká zranitelnost, kterou může zneužít útočník odposlouchávající komunikaci v dané síti. [14] [17] [18]

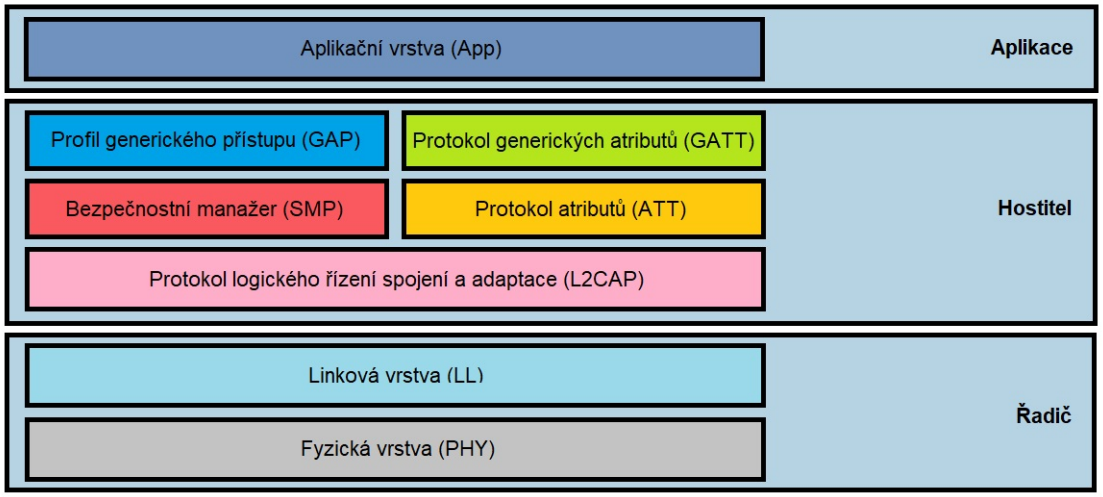
1.3 Bluetooth Low Energy (BLE)

Komunikační protokol BLE, také nazývaný jako Bluetooth Smart, vyvinula Bluetooth Special Interest Group (SIG) ve verzi Bluetooth 4.0 protokolu. Oproti klasickému

Bluetooth je spotřeba energie zařízení minimální a zároveň i cena je nízká. [14]

Specifikace BLE používá dvě hlavní části *Controller* a *Host* stejně jako klasické Bluetooth. Controller nebo-li Řadič je logická entita zodpovědná za fyzickou vrstvu (PHY) a linkovou vrstvu (LL). Host využívá funkcionality vyšších vrstev. Logical Link Control and Adaptation Protocol (L2CAP) zajišťuje přenášení dat mezi vyššími protokoly, segmentaci paketů a správu kvality pro vyšší protokoly (QoS - Quality of Services). [14] [19]

Generic Access Profile (GAP) udává obecné postupy související s vyhledáváním zařízení, navazování a ukončování spojení. Také definuje procedury závisející na různých úrovních zabezpečení. Security Manager (SMP) má na starosti správu párování, autentizaci, spojování a šifrování komunikace BLE. Attribute Protocol (ATT) specifikuje mechanismus na nalezení, čtení a zapisování atributů (hodnot) na dané zařízení. Generic Attribute Profile (GATT) definuje rámec založen na ATT pro zjišťování služeb. Host a Controller společně komunikují pomocí Host Controller Interface (HCI) standardu. [14] [19]



Obr. 1.4: Vrstvy BLE protokolu [20]

Protokol BLE má definované dva zabezpečovací režimy. *LE Security mode 1* je použit na linkové vrstvě a podporuje šifrování a podepisování dat pomocí šifrovacího standardu AES-128 a CCM režimu s různými úrovněmi autentizace. *LE Security mode 2* je aplikován na ATT vrstvě a zajišťuje jen podepisování dat. Tento mód je používán pouze pro nešifrovaná propojení. Níže rozepsáno v bodech: [21]

- Security mode 1
  1. Úroveň 1 - žádné zabezpečení
  2. Úroveň 2 - neautentizované párování s šifrováním (není provedena autentizace během výměny klíčů)

3. Úroveň 3 - autentizované párování s šifrováním (procedura párování, při které je vytvářen sdílený klíč, je prováděna pomocí zranitelného algoritmu)
4. Úroveň 4 - autentizované *LE Secure Conections* párování s šifrováním (díky užití eliptických křivek Diffie-Hellman s křivkou P-256 pro odvození sdíleného klíče je párování velmi dobře zabezpečené)

- Security mode 2
  1. Úroveň 1 - neautentizované párování s podepisováním dat
  2. Úroveň 2 - autentizované párování s podepisováním dat

Autentizační kód zprávy, ve specifikaci Bluetooth nazvaný *Message Integrity Check* (MIC), je používán pro podepisování dat. V Security mode 1 má délku 4 bajty a je spočítaný z obsahu zprávy (payload) a prvního bajtu hlavičky L2CAP. V Security mode 2 je dlouhý 8 bajtů, počítaný pomocí *block Cipher-based Message Authentication Code* (CMAC) a AES-128 ze zřetězených přenášených dat a 32 bitového čítače. Čítač je při každém podpisu zprávy inkrementován o 1. Tímto se zamezuje útoku formou opakovaného přehrání zprávy. [14] [22]

BLE používá tři sdílené klíče pro účel zabezpečení. Všechny tyto klíče jsou dlouhé 128 bitů. [14] [22]

- Long Term Key (LTK) - dlouhodobý klíč se používá při odvození klíče, který pak využívá linková vrstva
- Connection Signature Resolving Key (CSRK) - používán ATT vrstvou pro podepisování dat
- Identity Resolving Key (IRK) - používá se pro zajištění soukromí při komunikaci mezi zařízeními, tento klíč si mezi sebou stanoví po připojení

Zabezpečení celé komunikace nezáleží pouze na tom, jak jsou těžce prolomitelné použité šifry, ale rovněž jde o to, jakou cestou jsou sdílené klíče vytvořeny. Komunikační protokol BLE sdílené klíče stanovuje při párování. Používá jeden ze čtyř definovaných asociačních režimů pro vytvoření prvního dočasného klíče *Temporary Key* (TK) popsaných níže: [14] [22] [23] [24]

1. *Just Works* režim je možnost bez zabezpečení, jelikož TK má pevnou známou hodnotu. Útočník tedy může uskutečnit útok Man-In-The-Middle nebo pasivně odchytit proces párování a získat tak sdílené klíče. Tento režim je používán zařízeními, která nemají vstupní/výstupní schopnosti a nemohou žádat uživatele o potvrzení.
2. V *Out-of-Band* režimu se používá k přenesení TK jiná než dříve zavedená komunikační metoda, nebo jiný komunikační kanál. Je to tedy bezpečné samo o sobě. Ovšem jen velmi málo zařízení je vybaveno potřebným hardwarem pro fungování tohoto režimu.
3. Režim *Passkey Entry* se používá v případech, kdy jedno zařízení má vstup,

ale nemá žádnou výstupní zobrazovací schopnost. Uživatel může zadat identický šestimístný přístupový kód do obou zařízení, nebo v případě, kdy jedno zařízení je schopné nějakým algoritmem náhodně vygenerovat a zobrazit tento kód, tak už ho jen uživatel zadá do druhého zařízení. Tento režim je jednoduše prolomitelný Brute-Force útokem, jelikož je jen milion možných kombinací přístupového kódu. Ale od verze Bluetooth 4.2 je tento režim upraven a používá eliptické křivky Diffie-Hellmana (ECDH), což ho dělá mnohem bezpečnější.

4. *Numeric Comparsion* režim je používán, když obě zařízení mají display a vstupní schopnost pro hodnoty "true" a "false". Potvrzení uživatele je vyžadováno, aby zařízení vědělo, že druhé zařízení je legitimní a autentické a že je to právě to dané zařízení, ke kterému se uživatel snaží připojit. Tento režim nabízí omezenou ochranu proti útoku Man-In-The-Middle, ale pravděpodobnost úspěchu útočníka je jen 0,0001 %.

1.4 Penetrační testování

Obecně termín penetrační testování označuje aktivitu testera, který testuje nějakou aplikaci pomocí uskutečňování kybernetických útoků na systém. Účelem je ověřit, zda-li obsahuje zranitelnosti, kterých by se dalo zneužít. Tester se snaží proniknout zabezpečením díky proprietárním či volně dostupným nástrojům, které pak vypisují seznamy nalezených chyb, co mohou způsobit zranitelnost aplikace. Hlavním cílem penetračního testování je nalezení takových zranitelností v aplikaci, pomocí kterých by byl útočník schopný získat neoprávněný přístup do systému nebo aplikace, a tím tedy i přístup k citlivým datům.[25]

Stále se rozšiřující IoT prostor doplácí na to, jak se staví většina společností k vývoji levných IoT zařízení a neberou ohled na jejich zabezpečení. Tato zařízení mají mnoho zranitelností, které pak hackeři využívají k útoku na sítě, v nichž se zařízení nacházejí.[26]

1.4.1 BlackBox a WhiteBox testování

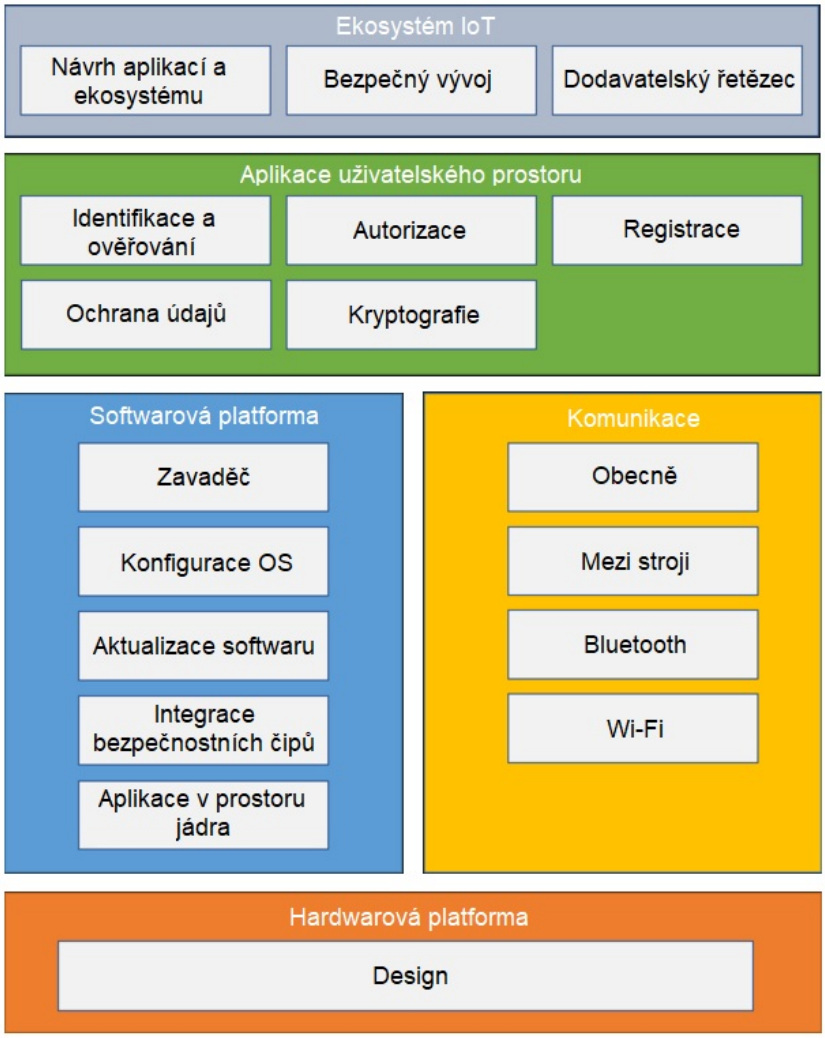
Testování Black Box je technika, při které není testerovi známo vnitřní fungování softwaru. Tester se zaměřuje pouze na vstup a výstup softwaru. Zatímco testování White Box je technika testování, při níž tester zná vnitřní fungování softwaru a může testovat jednotlivé fragmenty kódu. [27]

Cíl Black Box testování je zaměřen hlavně na program jako takový a kontroluje splnění požadavků na něj. Oproti tomu White Box se zaměřuje na zajištění funkcionality a efektivnosti interního kódu programu. Známým faktem je, že v případě Black Box se neví, jaké je vnitřní fungováni programu. Tedy tento test můžou provádět

bez znalosti struktury nebo programovacího jazyku. White Box testování tyto znalosti vyžaduje. Testování Black Box využívá metody, jako je analýza hraničních hodnot a odhadování chyb. Zatímco testování White Box používá metody jako testování toku řízení, testování toku dat a pokrytí příkazů. [27]

1.4.2 IoT Security Verification Standard - ISVS

Open Web Application Security Project (OWASP) je nezisková organizace zabývající se zabezpečením. Vytvořila i tento standard aby stanovila požadavky na zabezpečení aplikací internetu věcí (viz přehled Obr č. 1.5). Požadavky poskytované ISVS mohou být využity v mnoha stádiích během vytváření produktu včetně navrhování, vývoje a testování aplikací IoT. [28] [29]



Obr. 1.5: Přehled oblastí požadavků ISVS [28]

Bezpečnostní požadavky obsažené v tomto dokumentu pokrývají různé aspekty systémů a aplikací internetu věcí, včetně ekosystému internetu věcí, aplikací v uživatelském prostoru, softwarových platforem, komunikací a hardwarových platforem. Kromě toho se požadavky vztahují na mnoho fází vývoje produktu, včetně návrhu, vývoje a testování. OWASP IoT Security Verification Standard popisuje tři úrovně ověření zabezpečení. Každá úroveň se skládá ze seznamu požadavků mapovaných na schopnosti a funkce citlivé na zabezpečení. [10] [28] [29]

- ISVS úroveň 1
  Požadavky této úrovně cílí na poskytnutí ochrany proti útokům zaměřujícím se na software. Jedná se o útoky, které nezahrnují fyzickou manipulaci se zařízením. Tato úroveň poskytuje základní zabezpečení pro zařízení připojená v síti, kde by fyzické ohrožení daného zařízení nemělo velký dopad na zabezpečení. Jde o zařízení, která by neměla mít chráněnou IP adresu, nejsou na něm uložená žádná citlivá data a kompromitace tohoto zařízení v síti neumožňuje útočníkovi přístup do systému nebo jinému zařízení v síti. [28] [29]
  Zařízení spadající do této úrovně je například chytrá žárovka. Kompromitace této žárovky by útočníkovi nijak nepomohla k získání přístupu. Když nejsou na zařízení uložena žádná osobní data, není co získat. Pokud je autentizace a autorizace správně implementována na cloudové infrastruktuře, nejhorší scénář, kterého by mohl útočník dosáhnout, je podvrhnutí stavu kompromitované žárovky. [28] [30]
- ISVS úroveň 2
  Cílem požadavků druhé úrovně je poskytnutí ochrany proti útokům, které obcházejí software a směřují na hardware zařízení. U zařízení spadajících do této úrovně je potřeba se vyhnout jejich kompromitaci. IP adresa zařízení by již měla být v rozumné míře chráněna. Jedná se o zařízení, ve kterých je uložena nějaká forma citlivých informací. [28]
  Příklad zařízení spadající do této úrovně jsou chytré kamery, poplašné zabezpečovací systémy nebo lékařská zařízení, která shromažďují data z měření a odesílají lékaři. [28] [29] [30]
- ISVS úroveň 3
  Cílem této úrovně je poskytnutí požadavků pro zařízení, která nesmí být za každou cenu kompromitována. Na těchto zařízeních jsou uložena velmi citlivá data a jejich kompromitace může být zneužita. Mimo bezpečnostní požadavky stanovené v první a druhé úrovni se zde nacházejí techniky ochrany do hloubky. Snaží se zabránit reverznímu inženýrství a snahám o fyzickou manipulaci. [28] [30]
  Příkladem zařízení této úrovně jsou kryptoměnové peněženky, připojená vozidla nebo také lékařské implantáty. [28]

Požadavky na ekosystém IoT

Tato oblast požadavků tvoří spojení mezi připojeným zařízením do sítě a jeho okolím v ní.

Aby byla zajištěna bezpečnost vytvořeného softwaru, build (proces sestavení) musí být proveden v bezpečném prostředí, kde se ověří integrita a autentičnost všech komponent. Kód musí být napsán tak, aby používal osvědčené bezpečnostní postupy a zkompilován za použití nejlepších dostupných možností zabezpečení. [28] [30]

Změny v konfiguracích systému musí využívat vhodné možnosti protokolování a monitorování, aby poskytovaly záznamy o změnách v zabezpečení. Podrobné zprávy o událostech jsou mnohdy nápomocné. Když jsou však příliš podrobné a obsahují citlivé informace, představují bezpečnostní rizika. [28] [30]

Kontrolní seznam zabývající se touto částí je k dispozici v standardu ISVS [28].

Požadavky na aplikace v uživatelském prostoru

Pro zabezpečení přístupu je nutná autentizace a autorizace. Mezi relevantní požadavky patří jedinečná silně zabezpečená identita, rozdělení uživatelských rolí a koncept nejmenšího oprávnění. Autentizace je akt stanovení nebo potvrzení identity, ať už osoby či zařízení, jako autentické. Je to základ pro přesvědčení, že tvrzení získané od osoby nebo zařízení jsou správná a jsou odolná proti předstírání identity. Mezi další nezbytné vlastnosti patří předejití obnovení nebo zachycení autentizačních údajů jako jsou hesla, API klíče nebo také soukromé klíče. Autorizace je akt stanovení nebo potvrzení přístupových práv ke zdrojům nebo akcím splňujícím zásady bezpečného přístupu. [28] [30]

Ochrana citlivých dat včetně přihlašovacích údajů a správné zacházení se soukromými informacemi je nezbytné k zajištění bezpečného užívání systémových prostředků, jako jsou soubory obsahující kód nebo data obsažená v paměti. Spousta požadavků v této části je implementována pomocí kryptografie. [28] [30]

Kontrolní seznam, týkající se identifikace a autentizace, autorizace, ochraně dat a kryptografie, je k nalezení ve standardu ISVS [28].

Požadavky na software

Bootloader nebo-li zavaděč je první část kódu, který se spustí během spouštěcího procesu zařízení. Výrobce firmware je zodpovědný za správnou konfiguraci bootloaderů. V případě špatného nastavení by mohly vzniklé zranitelnosti zničit celé zabezpečení zařízení. Požadavky této části zajišťují důvěryhodnost bootování pomocí ověřování kryptografických podpisů na načteném kódu. Během tohoto procesu je zakázán přístup k paměti, shellu a dalšímu ladění. [28] [30]

Operační systém a zejména jeho jádro jsou pro zabezpečení zařízení zásadní, jelikož běží v privilegovaném režimu a implementují důležité funkce zařízení, včetně mnoha základních zabezpečení. Vyžaduje to nejlepší bezpečnostní postupy pro operační systém, konfiguraci jádra a zpevnění (největší možné snížení počtu vykonávaných funkcí systémem, kvůli možným zranitelnostem). [28] [30]

Linux je jedním z nejpoužívanějších operačních systémů v IoT. Má velmi rozšířené zabezpečovací funkce, včetně izolačních mechanismů podporovaných jmennými prostory, cgroups a další zabezpečovací moduly pro řízení přístupu. Tyto izolační mechanismy by měly být využity při konfiguraci a nasazování aplikací třetích stran ke spuštění v kontejneru. [28] [30]

Aktualizace a údržba softwaru zařízení je také velmi důležitá pro zabezpečení produktu. Zařízení musí mít implementovány aktualizační systémy, které využívají osvědčené bezpečnostní postupy. Zajišťují, aby zařízení používalo jen kryptograficky podepsaný software neobsahující žádnou známou zranitelnost. Procesy správy opravných balíčků (patchů) a zranitelností zajišťují nasazení nejnovějších dostupných verzí sestavení (tzv. buildů) obsahující upstream zabezpečovací balíčky, odkazující na původní autory softwaru, které chrání koncové uživatele před kompromitováním. [28] [30]

Bezpečná konfigurace a integrace hardwarových bezpečnostních čipů do softwarových platforem umožňuje zařízení používat kryptograficky potvrzenou identitu vypálenou do čipu během výroby. Tyto bezpečnostní čipy přidávají další funkce a poskytují privilegovaná úložiště určená pro klíče. [28] [30]

Kontrolní seznam zaměřený na bootloader, konfiguraci operačního systému Linux, softwarové aktualizace, integraci zabezpečovacího čipu a prostoru jádra je dostupný ve standardu ISVS [28].

Požadavky na komunikaci

Aby různé strany v síti mohly důvěřovat obsahu komunikace, je potřeba zajistit autentičnost stran, integritu dat proti škodlivým změnám a zamezení úniku informací. V praxi je to řešeno nasazením aktuálních komunikačních protokolů s nakonfigurováním jejich bezpečnostních prvků včetně kryptografie. Vzhledem k tomu, že se průmyslové směrnice na zabezpečení TLS (Transport Layer Security), Bluetooth nebo Wi-Fi často mění, měly by být proto konfigurace pravidelně kontrolovány, aby bylo zabezpečení vždy účinné: [28] [30]

- Vždy by mělo být použito TLS nebo ekvivalentní silné šifrování a autentizace, bez ohledu na citlivost přenášených dat.
- Mezi další bezpečnostní postupy patří autentizace na základě připínání certifikátu a vzájemné autetizaci.

- V konfiguraci by mělo být nastavené preferované pořadí algoritmů a šifer používaných pro komunikaci a zakázané zastaralé nebo známé nezabezpečené algoritmy a šifry.
- Mělo by být nastaveno nejlepší možné dostupné zabezpečení.

Kontrolní seznam týkající se komunikace obecně, mezi zařízeními a nejpoužívanějších komunikačních protokolů je také k nalezení ve standardu ISVS [28].

Požadavky na hardware

Oproti softwaru je kompromitace hardwaru mnohem komplikovanější a nákladnější. Hardwarové zabezpečení tedy může poskytnout dobrý základ pro zabezpečení vestavěných zařízení. Ale v případě, kdy hardware obsahuje zadní vrátka nebo nezdokumentované funkce ladění, může být ohrožena bezpečnost celého zařízení bez ohledu na to, jaká byla přijata bezpečnostní opatření na jiných vrstvách. [28].

Účelem požadavků v této oblasti je zajištění, aby byl hardware nakonfigurován nejbezpečnějším možným způsobem. To zahrnuje deaktivaci nebo zabezpečení ladících rozhraní. Zapnutí všech existujících alarmů a senzorových mechanismů pro zabránění nedovolené manipulaci. Užití ochrany hardwaru proti klonování, jako jsou například pojistky OTP (One-Time Programmable) a použití MMU (Memory Management Unit) pro izolaci procesů. [28] [30]

Kontrolní seznam pro tuto část je taktéž k nalezení ve standardu ISVS [28].

1.5 Kontrolní seznam

1.5.1 Chytrá žárovka - BLE

Na tomto zařízení zpravidla nejsou uložena žádná citlivá data a útočníkovi neumožní přístup do systému nebo k jinému zařízení v síti. Podle standardu ISVS tedy toto zařízení spadá do skupiny Úrovně 1. [28]

Obecné požadavky

- (ISVS 4.1.1) Ověřte, zda komunikace s ostatními součástmi ekosystému IoT (včetně snímačů, brány a podpůrného cloudu) probíhá prostřednictvím zabezpečeného kanálu, v němž je zaručena důvěrnost a integrita dat a v němž je do komunikačního protokolu zabudována ochrana proti útokům typu replay. [28]
- (ISVS 4.1.2) Pomocí aktuálních testovacích nástrojů TLS ověřte, zda jsou povoleny pouze silné sady šifer, přičemž nejsilnější sada šifer je nastavena jako preferovaná. [28]

- (ISVS 4.1.3) Ověřte, zda v případě použití protokolu TLS zařízení kryptograficky ověřuje certifikát X.509. [28]

Machine-to-Machine požadavky

- (ISVS 4.2.1) Ověřte, zda je nešifrovaná komunikace omezena na data a instrukce, které nejsou citlivé povahy. [28]
- (ISVS 4.2.2) Ověřte, zda zprostředkovatelé MQTT umožňují přihlášení k odběru témat a publikování zpráv pouze autorizovaným zařízením IoT. [28]
- (ISVS 4.2.3) Ověřte, zda jsou pro ověřování transakcí MQTT upřednostňovány certifikáty před nativním uživatelským jménem a heslem. [28]

Buetooth

- (ISVS 4.3.1) Ověřte, zda je párování a zjišťování v zařízeních Bluetooth blokováno, s výjimkou nezbytných případů. [28]
- (ISVS 4.3.2) Zkontrolujte, zda kódy PIN nebo PassKey nejsou snadno uhodnutelné (např. nepoužívejte 0000 nebo 1234). [28]
- (ISVS 4.3.3) Ověřte, zda zařízení používající starší verze Bluetooth s povolenými jednoduchými režimy ověřování vyžadují pro párování kód PIN. [28]
- (ISVS 4.3.4) Ověřte, zda je u moderních verzí Bluetooth vyžadováno alespoň 6 číslic pro ověření SSP (Secure Simple Pairing) ve všech verzích kromě "Just Works". [28]
- (ISVS 4.3.5) Ověřte, že šifrovací klíče mají maximální velikost, kterou zařízení podporuje, a že tato velikost je dostatečná pro dostatečnou ochranu informací přenášených prostřednictvím připojení Bluetooth. [28]
- (ISVS 4.3.6) Zkontrolujte, zda je použita nejbezpečnější dostupná metoda párování Bluetooth. Ověřte, zda se v závislosti na možnostech komunikujícího zařízení používají metody párování mimo pásmo (OOB), číselné porovnání nebo zadání přístupového klíče. [28]
- (ISVS 4.3.7) Ověřte, zda je použit nejsilnější režim a úroveň zabezpečení Bluetooth, které zařízení podporuje. Například u zařízení Bluetooth 4.1 by měl být použit režim zabezpečení 4, úroveň 4, který zajistí ověřené párování a šifrování. [28]

1.5.2 IoT zařízení pracující na frekvenci 433MHz

Takové zařízení může být použito například jako prvek chytrého domu. Zařízení reprezentující tento prvek je kupříkladu dveřní čidlo nebo detektor pohybu pracující

s ústřednou ovládající osvětlení. Podle standardu ISVS by tedy spadalo toto zařízení do skupiny úrovně 2. [28]

V praktické části byla prezentována zranitelnost komunikace při těchto zařízeních. V zakoupeném kitu nebylo přítomné přímo takové zařízení. Přistoupilo se k jeho náhražce jinými, jednoduššími. Tyto zařízení simulují jeho možnou komunikaci a zranitelnost (viz 2.4 Reverzace rádiových signálů).

- (ISVS 4.2.1) Ověřte, zda je nešifrovaná komunikace omezena na data a instrukce, které nejsou citlivé povahy. [28]

1.5.3 Zařízení komunikující v síti ZigBee

Opět jako v předešlé podkapitole, se v praktické části nahradilo IoT zařízení komunikující pomocí Zigbee protokolu. K tomuto účelu bylo použito opět Arduino. Bylo naprogramované jako zdroj dat. Ta byla opakovaně vysílána pomocí Xbee modulu.

ZigBee

- (ISVS 4.5.1) Ověřte, zda se pro nové aplikace používá Zigbee verze 3.0. [28]
- (ISVS 4.5.2) Ověřte, zda je zvolena vhodná architektura zabezpečení Zigbee (centralizovaná nebo distribuovaná) v závislosti na požadavcích na úroveň zabezpečení aplikace a modelu hrozeb. Centralizovaná architektura obecně nabízí vyšší zabezpečení na úkor flexibility. [28]
- (ISVS 4.5.3) Ověřte, zda je použit nejbezpečnější způsob připojení k síti Zigbee v závislosti na zvolené architektuře zabezpečení. Například pro centralizovanou architekturu použijte out-of-band instalační kódy. Pro distribuovanou použijte předem nakonfigurované spojovací klíče. [28]
- (ISVS 4.5.4) Ověřte, zda se pro připojení k síti nepoužívá výchozí předkonfigurovaný globální spojovací klíč (tj. ZigbeeAlliance09), s výjimkou případů, kdy je to výslovně vyžadováno z důvodů kompatibility a kdy byla zohledněna související rizika. [28]
- (ISVS 4.5.5) Ověřte, že k aktivaci režimu párování pro připojující se uzly a Zigbee Trust Center nebo router je vyžadována interakce uživatele. Zařízení by měla automaticky opustit režim párování po předem definované krátké době, a to i v případě neúspěšného párování. [28]
- (ISVS 4.5.6) Ověřte, zda je síťový klíč náhodně vygenerován (například při počátečním nastavení sítě). [28]

1.5.4 Chytrá zásuvka - Wi-Fi

Toto zařízení, stejně jako chytrá žárovka, v sobě nemá uložena žádná citlivá data. Proto je také zařazeno do skupiny úrovně 1 standardu ISVS.

Identifikace a autentizace

- (ISVS 2.1.1) Ověřte, zda lze všechny formy uživatelů a účtů v ekosystému internetu věcí jednoznačně identifikovat. [28]
- (ISVS 2.1.2) Ověřte, zda lze všechna připojená zařízení v rámci ekosystému internetu věcí jednoznačně identifikovat, včetně připojení ke cloudu, rozbočovačům i dalším zařízením (např. senzorům). [28]
- (ISVS 2.1.3) Ověřte, zda je v celém ekosystému internetu věcí zavedeno silné ověřování uživatelů a zařízení. [28]
- (ISVS 2.1.4) Ověřte, zda schémata ověřování uživatelů, služeb a zařízení sdílejí společný rámec centrálně spravovaný v ekosystému internetu věcí. [28]
- (ISVS 2.1.5) Ověřte, že hesla používaná pro ověřování uživatelů mají alespoň 12 znaků. [28]
- (ISVS 2.1.6) Ověřte, že hesla používaná pro ověřování uživatelů může uživatel změnit a že funkce změny hesla vyžaduje aktuální a nové heslo uživatele. [28]
- (ISVS 2.1.7) Ověřte, zda jsou hesla používaná pro ověřování zařízení dostatečně dlouhá a složitá. [28]
- (ISVS 2.1.8) Ověřte, zda mohou oprávnění správci nebo koncoví uživatelé změnit výchozí pověření uživatele nebo zařízení. [28]
- (ISVS 2.1.9) Ověřte, zda autentizační údaje pro uživatele, zařízení nebo služby nejsou pevně zakódovány ve firmwaru nebo ekosystému. [28]
- (ISVS 2.1.10) Ověřte, zda jsou poskytnuté pověření pro ověřování zařízení jedinečné pro každé zařízení. [28]

Autorizace

- (ISVS 2.2.1) Ověřte, zda účty systému IoT napříč uživateli, službami a zařízeními sdílejí společný autorizační rámec. [28]
- (ISVS 2.2.2) Ověřte, zda zařízení prosazují koncept nejmenších oprávnění omezením aplikací a služeb spouštěných jako root nebo správce. [28]
- (ISVS 2.2.3) Ověřte, zda je vlastnictví ověřeno při registraci a v rámci vyřazení z provozu, když se zařízení přesouvají mezi účty (např. přeprodej zařízení, leasing a pronájem). [28]

Ochrana dat

- (ISVS 2.3.1) Ověřte, zda jsou citlivé informace, jako jsou osobní identifikační údaje (PII) a pověření, bezpečně uloženy pomocí silného šifrování na ochranu před únikem dat a kontroly integrity na ochranu před neoprávněnou změnou. [28]
- (ISVS 2.3.2) Ověřte, že v případě vyřazení zařízení z provozu nebo změny vlastníka lze ze zařízení odstranit všechny citlivé informace, jako jsou údaje PII a pověření. [28]
- (ISVS 2.3.3) Ověřte, zda je zařízení v případě vyřazení z provozu nebo změny vlastníka označeno v centrálně spravované databázi v ekosystému pro účely auditu. [28]

Kryptografie

- (ISVS 2.4.1) Ověřte, zda jsou kryptografická tajemství a klíče pro každé zařízení jedinečné. [28]
- (ISVS 2.4.2) Ověřte správné používání kryptografie. Měly by se používat pouze standardní a silné algoritmy s odpovídající velikostí klíče a bezpečnou implementací. [28]

Obecné požadavky

Tato skupina obecných požadavků je identická jako výše v části "Chytrá žárovka - BLE".

Wi-Fi

- (ISVS 4.4.1) Ověřte, že připojení Wi-Fi je zakázáno, pokud není vyžadováno jako součást funkce zařízení. Zařízení, která nepotřebují síťové připojení nebo která podporují jiné typy síťového připojení, například Ethernet, by měla mít rozhraní Wi-Fi vypnuté. [28]
- (ISVS 4.4.2) Zkontrolujte, zda je k ochraně komunikace Wi-Fi použito WPA2 nebo vyšší. [28]
- (ISVS 4.4.3) Ověřte, zda je v případě použití WPA použito šifrování AES (režim CCMP). [28]
- (ISVS 4.4.4) Zkontrolujte, zda se k navázání připojení Wi-Fi mezi zařízeními nepoužívá funkce WPS (Wi-Fi Protected Setup). [28]

2 Praktická část bakalářské práce

Pro splněni praktické části byl zakoupen kit na Fakultě elektrotechniky a komunikačních technologií VUT v Brně. Použitá zařízení z IoT Exploitation Learning Kitu jsou uvedeny na začátku jednotlivých kapitol. Praktická část je rozdělena do čtyř částí, přičemž se každá z nich věnuje jinému IoT zařízení. Každá táto část je rozdělena na dvě, a to popis BlackBox, případně WhiteBox testovaní a samotný popis manuálního testovaní testerem. Celkově byly v této kapitole předvedeny čtyři zranitelnosti. Všechny následující exploitace jsou prováděny v systému Linux Ubuntu verze 20.04.

2.1 Black Box testování chytré LED žárovky

Je inteligentní žárovka v domácnosti schopna odolat útoku při zneužití odchycené komunikace mezi chytrým telefonem a zařízením samotným?

Popis testu

Úkolem testu je ověření zranitelnosti IoT zařízení, v tomto případě inteligentní BLE LED žárovky, a možnosti jejího ovládání z libovolného zařízení odesíláním instrukcí na její adresu bez mobilní aplikace. Musí se dodržet formáty řetězců, které je žárovka schopna zpracovat. Proto je potřebné zjistit, kam se posílají tyto řetězce a odhalit v jakém formátu. Správný formát řetězce je vhodné zjistit z odchycené komunikace, jelikož prolomení pomocí brute force by bylo časově extrémně náročné. Důležitá část se týká samotného zjištění, kam tyto se tyto pakety adresují do konkrétního zařízení.

Jak test provést manuálně

Zjistěte, jakou MAC adresu má žárovka. Během testu ověřte, zda je možné odchytit (např. pomocí zařízení Ubertooth One) komunikaci a odhalit, na jakou adresu se zasílají pakety s instrukcemi pro žárovku. Získejte z odchycené komunikace řetězce pro několik barev. Následně se je pokuste posílat zpět jako novou instrukci, která byla původně uživatelem odeslána pomocí mobilní aplikace ovládající žárovku. Pokud žárovka reaguje na odeslané instrukce, pokuste se analyzovat z odchycených řetězců, kde se nachází byty prezentující hodnoty RGB a jejich vazbu na zbytek řetězce.

Testem lze odhalit tyto zranitelnosti

- Nešifrovaná komunikace
- Nevyžadování autentizace
- Možné ovládání žárovky neoprávněnou osobou

2.2 BLE LED žárovka - exploitace

V této kapitole je předvedeno, jak se odchytává komunikace mezi mobilním telefonem a chytrou žárovkou. Následně byla provedena analýza odchycených paketů a zjištěna možnost využití získaných informací k převzetí kontroly nad zařízením.

2.2.1 Použitá zařízení

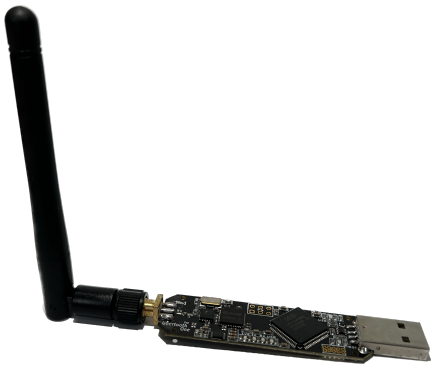
- Chytrá BLE LED žárovka (Obr. č. 2.1)

Obr. 2.1: Chytra BLE LED žárovka Govee

  Byla exploitována žárovka od společnoti Govee model H6001.
- Bluetooth 4.0 USB dongle adaptér (Obr. č. 2.2)

Obr. 2.2: Bluetooth USB adaptér

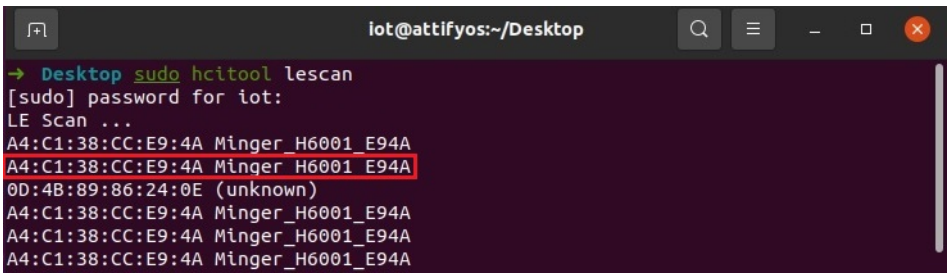
  Adaptér umožňující interakci systému s BLE zařízeními. Bude použit ke zjištění adresy zařízení a také ke čtení a úpravě BLE atributů cílového zařízení.
- Ubertooth One (Obr. č. 2.3)
  Ubertooth je open-source 2,4 GHz nástroj vhodný na experimentování s Bluetooth. Umí zachytávat i vysílat pakety, identifikovat a následovat přeskakování kanálů cílového zařízení.
- Mobilní telefon
  Ten je potřebný pro instalaci aplikace na ovládání žárovky.



Obr. 2.3: Ubertooth One

2.2.2 Průzkum a odposlech

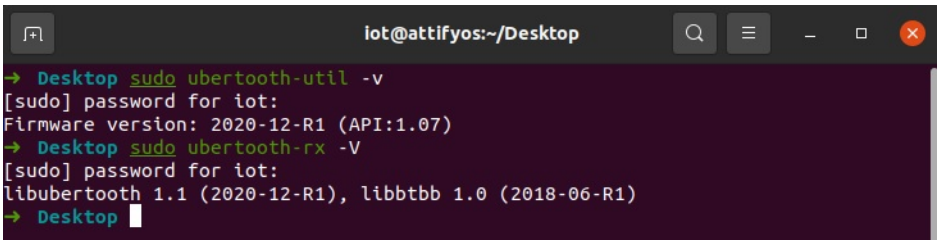
Nejprve bylo potřeba nainstalovat aplikaci (Govee Home) pro chytrou žárovku do mobilního telefonu a její spárování s aplikací. Pro porozumění příkazů, které posílá aplikace chytré žárovce, je potřebné odchytit BLE komunikaci mezi mobilním telefonem a cílovým zařízením. Prvním krokem je zjištěna adresa chytré žárovky použitím níže zobrazeného komandu `sudo hcitool lescan`. Všechny příkazy se zadávají do příkazového řádku tzv. Terminál, který se dá otevřít pravým kliknutím na plochu a výběrem "open in terminal".



Obr. 2.4: Zjištění adresy zařízení

Z obrázku č. 2.4 je zřejmé, že adresa žárovky je A4:C1:38:CC:E9:4A. Pomocí toho je možné pokračovat v odchytávání paketů použitím BLE odposlechového zařízení. Tímto zařízením je již výše zmíněný Ubertooth One. Ten pro svou funkci potřebuje Ubertooth-utils, což je univerzální nástroj pro provádění obecně užitečných věcí s Ubertooth.

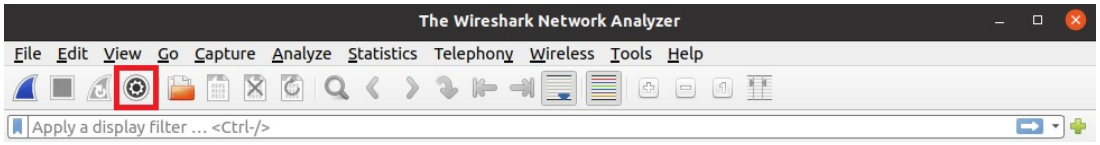
Po připojení Ubertooth One do systému se může zkontrolovat verze firmwaru zařízení a verze knihovny pro dekódování BLE paketů spuštěním následujících příkazů v terminálu (Obr. č. 2.5).



Obr. 2.5: Kontrola verze firmwaru a knihovny

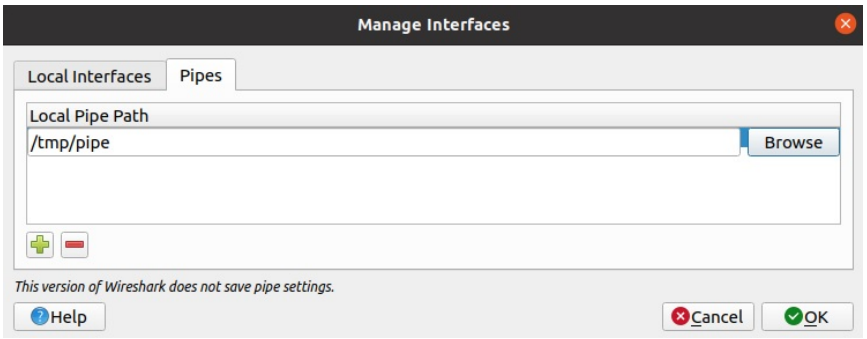
V případě, že není firmware nebo knihovna aktuální, je možné ji aktualizovat podle návodu níže v kapitole Řešení případných problémů. Tento postup je vhodné apikovat pro zajištění správného fungování s novými zařízeními. Jakmile je firmware i knihovna aktuální, může se přejít na samotný odchyt BLE komunikace. Aby se daly zobrazovat pakety v reálném čase, musí se vytvořit takzvaný pipe interface, příkazem "sudo mkfifo /tmp/pipe". Ten se používá jako místo, kam se bude ukládat zachycený provoz sítě.

V této chvíli je potřebné si spustit Wireshark a nakonfigurovat jej. V něm se zvolilo rozhraní na odchycení paketů, kliknutím na tlačítko Capture Interfaces, jak je znázorněno níže na obrázku č. 2.6.



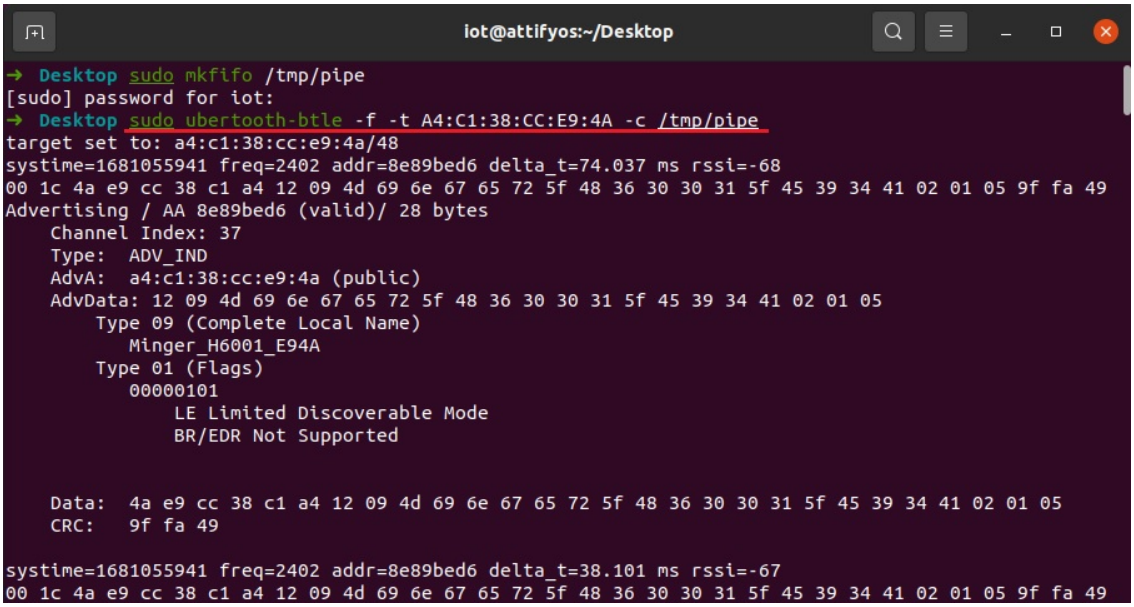
Obr. 2.6: Wireshark Konfigurace

Po kliknutí na tlačítko se vybralo Manage Interfaces > Pipes. Po té v záložce Pipes se přidala destinace /tmp/pipe a vše se potvrdilo (Obr. č. 2.7).



Obr. 2.7: Přidání destinace

Před samotným startem odchytávání komunikace se na mobilním telefonu vypnulo bluetooth nebo se odpojil od chytré žárovky, kvůli korektnímu spuštění odposlouchávajícího zařízení. Jakmile bylo zařízení odpojeno, mohl se spustit nástroj ubertooth-btle na započetí odposlechu (Obr. č. 2.8) se značkami -f pro sledování BLE komunikace, -t pro specifikaci adresy odposlouchávaného zařízení a -c pro místo ukládání odchycené komunikace.

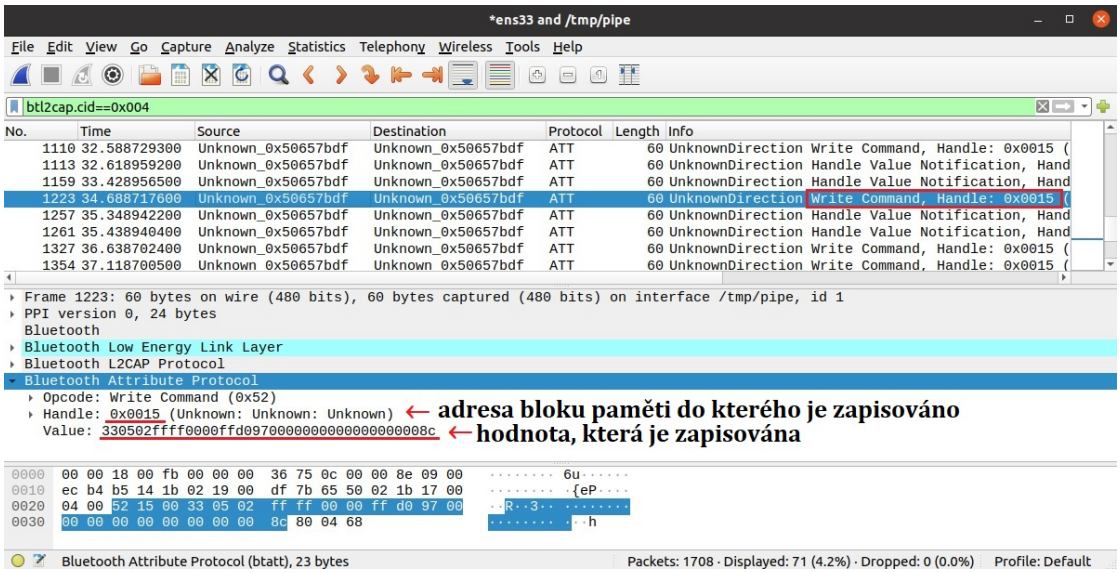


Obr. 2.8: Start odposlechu

V této chvíli se začal objevovat výpis paketů v terminálu s dotazováním chytré žárovky.

Ve Wiresharku se začaly ukazovat všechny odchycené BLE pakety. Nicméně většina těchto paketů jsou dotazovací a nejsou v tomto případě nijak důležité. Ve Wiresharku aplikujeme filtr pro zobrazení jen ATT paketů použitím "btl2cap.cid=0x004" (obr. č. 2.9). Následně proběhlo připojení přes mobilní aplikaci k žárovce a měnily se přes ni barvy žárovky, jas, teplota bílé a další možnosti jejího nastavení.

Po prozkoumání zachycené komunikace bylo zjištěno, že se data pro interakci s žárovkou ukládají do místa v paměti 0x0015.



Obr. 2.9: Filtrování odchycených paketů ve Wiresharku

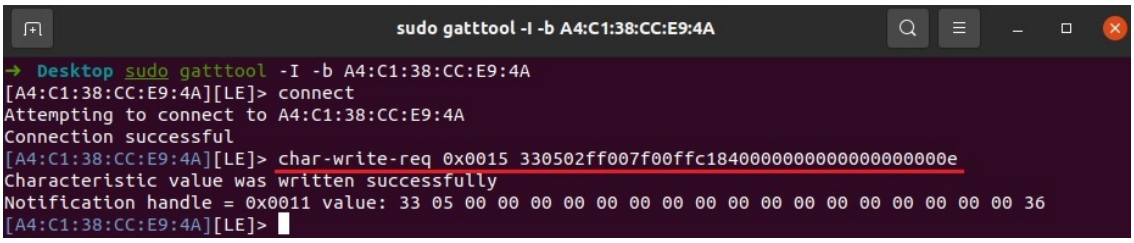
2.2.3 Převzetí kontroly nad žárovkou

Na řadě bylo převzetí kontroly. Byly použity informace obsažené v odchycených paketech. Je možné vepisovat požadované hodnoty dat do takzvaných charakteristik, které kontrolují stav žárovky.

Pro tento účel byl použit nástroj Gatttool. To je nástroj umožňující manipulaci s atributy BLE zařízení. Nejdříve se navázalo spojení s chytrou žárovkou.

Obr. 2.10: Navázání spojení pomocí nástroje Gatttool

Charakteristiky je možné prozkoumat použitím příkazu `characteristics`. Z vypsaných charakteristik z obrázku č. 2.10 je zajímavá jen 0x0015, do které budou následně zapisovány jednotlivé hodnoty. Pro zapisování hodnot se používá příkaz char-write-req ve tvaru: "char-write-req adresa hodnota", jak je zobrazeno níže na obrázku č. 2.11.



Obr. 2.11: Změna barvy žárovky

Jakmile byla hodnota zapsána, chytrá žárovka změnila barvu. Dalším možným postupem je používání variací jednotlivých řetězců odchycených z komunikace. V této práci byl vytvořen seznam řetězců (příloha A), kde jsou popsány jednotlivé kategorie funkcí. Ukázka (Obr. č. 2.12) jedné kategorie a to konkrétně změny teploty bílé barvy.

```
69 ---------------------
70 TEPLOTA bile
71
72        tepla
73                330502ffffff01ff8d0b00000000000000000b3      max
74                330502ffffff01ff98360000000000000000009b
75                330502ffffff01ffb16500000000000000000e1
76                330502ffffff01ffd0970000000000000000072
77                330502ffffff01ffdbba0000000000000000054
78                330502ffffff01ffead300000000000000000c
79                330502ffffff01fff2ec000000000000000002b
80                330502ffffff01fef9ff0000000000000000032
81                330502ffffff01f3f2ff0000000000000000034
82                330502ffffff01e6ebff0000000000000000038
83                330502ffffff01dde6ff000000000000000000e
84                330502ffffff01d6e1ff0000000000000000002      max
85        studena
```

Obr. 2.12: Ukázka přílohy A

2.2.4 Řešení případných problémů

Aktualizace firmware

Prvně je potřeba stáhnout poslední verzi firmware Ubertooth, která je volně dostupná na internetu. Poté extrahovat archiv a přejít do adresáře "ubertooth-one-firmware-bin". Příkaz `ubertooth-dfu` se pak může spustit následovně:

```
$ ubertooth-dfu -d bluetooth\_rtx.dfu -r
```

Zařízení automaticky přejde do DFU (Device Firmware Update) módu a provede nahrátí firmwaru.

Pokud se na konci zobrazí zpráva "control message unsupported", znamená to, že resetování zařízení nebylo úspěšné. Může se to vyřešit spuštěním příkazu `ubertooth-util -r` nebo se zařízení Ubertooth odpojí z USB a následně zase připojí.

Aktualizace knihovny

Před tím, než se začne s aktualizací knihovny, je potřebné mít nainstalované nějaké prerekvizity. Mnohé z nich jsou k dispozici v repozitářích balíčků operačního systému. V tomto případě kdy je využit Linux Ubuntu 20.04 spustíme následující příkaz:

```
$ sudo apt install cmake libusb-1.0-0-dev make gcc g++ lib
bluetooth-dev wget \pkg-config python3-numpy python3-qtpy
python3-distutils python3-setuptools
```

Dále se vytvoří knihovna základního pásma Bluetooth (libbtbb), aby nástroje Ubertooth mohly dekódovat pakety Bluetooth:

```
$ wget https://github.com/greatscottgadgets/libbtbb/archi
ve/2020-12-R1.tar.gz -O libbtbb-2020-12-R1.tar.gztar -xF
libbtbb-2020-12-R1.tar.gz
$ cd libbtbb-2020-12-R1
$ mkdir build
$ cd build
$ cmake ..
$ make
$ sudo make install
$ sudo ldconfig
```

2.3 Black Box testování odchycené radiové komunikace

Popis testu

Úkolem tohoto testu je ověření, zda jsou přenášená data zabezpečena a nemůže tak docházet k neoprávněnému čtení.

Jak provést test manuálně

Zjistěte, na jaké frekvenci je zařízení aktivní. Zachyťte radiový provoz na dané frekvenci. Proveďte analýzu zachyceného signálu. Zkoumejte strukturu dat a identifikujte vzorce, které by mohly vést k určení modulačního schéma a následně ještě k určení datového formátu. Na základě analýzy se pokoušejte rekonstruovat původní data.

Testem lze odhalit tyto zranitelnosti

- Nešifrovaná komunikace

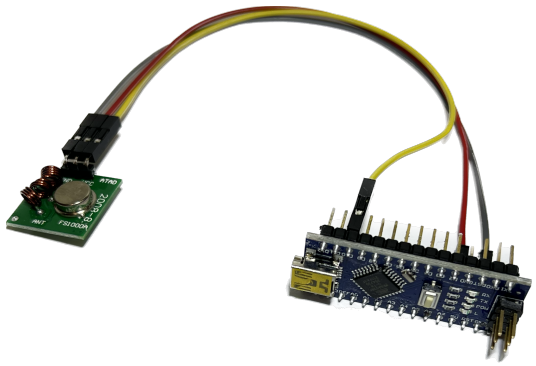
2.4 Reverzace rádiových signálů

V této kapitole je předvedeno, jak lze odchytit rádiové signály pomocí zařízení RTL-SDR a jejich reverzní převedení do původních dat.

2.4.1 Použité Prostředky

Hardware

- Arduino Nano a 433 MHz vysílač (Obr. č. 2.13)



Obr. 2.13: Arduino Nano a 433MHz vysílač

Tato dvě zařízení spolu simulují komunikační prostředí, například jednoduchých senzorů teploty, vlhkosti nebo třeba osvětlení. Také se dají použít i na poněkud citlivějších místech, a to například při zabezpečení domácnosti dveřními senzory, detektory pohybu nebo systémy pro vzdálené ovládání zabezpečení.

- RTL-SDR (Obr. č. 2.14)

Obr. 2.14: RTL-SDR s anténou

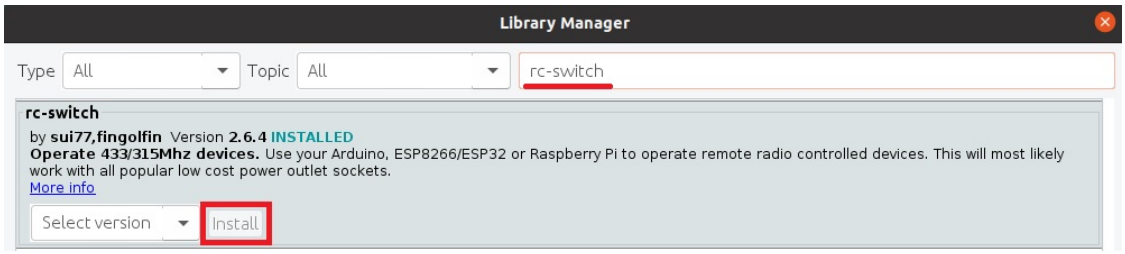
Software

- GQRX
- GNURadio
- Arduino IDE
- Audacity

2.4.2 Příprava zařízení

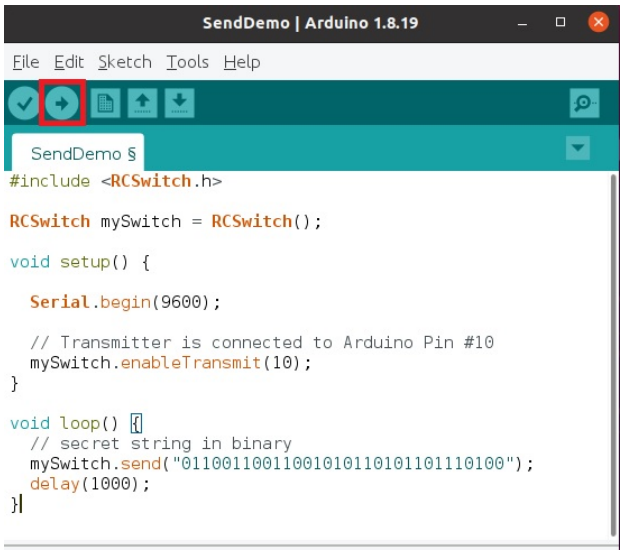
V první řadě bylo zapotřebí naprogramovat Arduino Nano jednoduchým programem, který bude představovat vysílání potenciálně citlivých dat. Naprogramování se provedlo programem Arduino IDE. Využila se knihovna RC-Switch, ze které byl vybrán příklad programu a modifikován pro tuto úlohu. Knihovna RC-Switch se získá stáhnutím v záložce "Tools > Manage Libraries"v menu Arduino IDE, jak je ukázáno na obrázku č. 2.15.

Použil se program SendDemo.ino, který je k nalezení v záložce "File > Examples > rc-switch". Tento program používá knihovnu RC-Switch pro připojení k 433 MHz vysílači na pinu D10 Arduina s přenosovou rychlostí 9600.



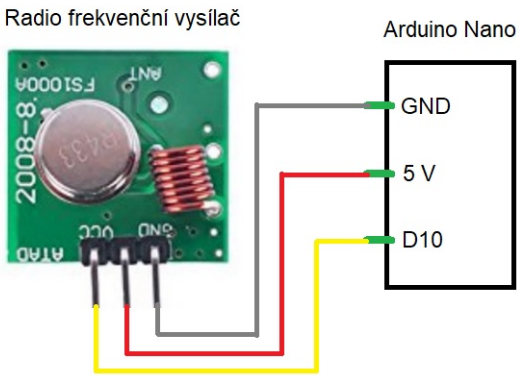
Obr. 2.15: Stažení knihovny RC-Switch

Vysílaná data byla "01100110011001010110101101110100" s klíčováním amplitudovým posunem (Amplitude Shift Keying) každou 1 sekundu. Výsledný program by měl vypadat jako na obrázku č. 2.16 a nahrát se do zařízení zvýrazněným tlačítkem "Upload". Jakmile byl program úspěšně přenesen, zařízení se odpojilo z USB.



Obr. 2.16: Program - Arduino Nano

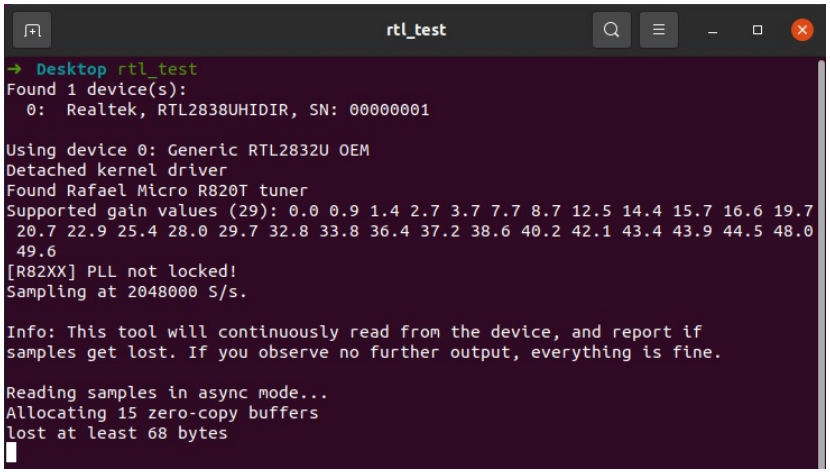
Dále se pokračovalo s připojením vysílače k Arduinu, které se provedlo podle obrázku č. 2.17. Tímto bylo zařízení připraveno k vysílání.



Obr. 2.17: Zapojení Arduino Nano + 433 MHz vysílač

2.4.3 Identifikace frekvence s RTL-SDR

V této fázi bylo potřebné zprovoznit zařízení RTL-SDR. Po připojení do systému je možné provést kontrolní test použitím nástroje `rtl_test` a zjistit, zda je zařízení schopné zachytávat pakety a nemá velkou ztrátovost paketů (Obr. č. 2.18).

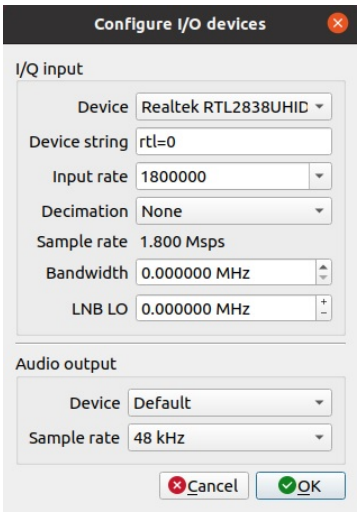


Obr. 2.18: Test zařízení RTL-SDR

Pokud nedochází ke ztrátám paketů, tedy není žádný další výstup `rtl_testu`, indikuje, že zařízení RTL-SDR funguje správně.

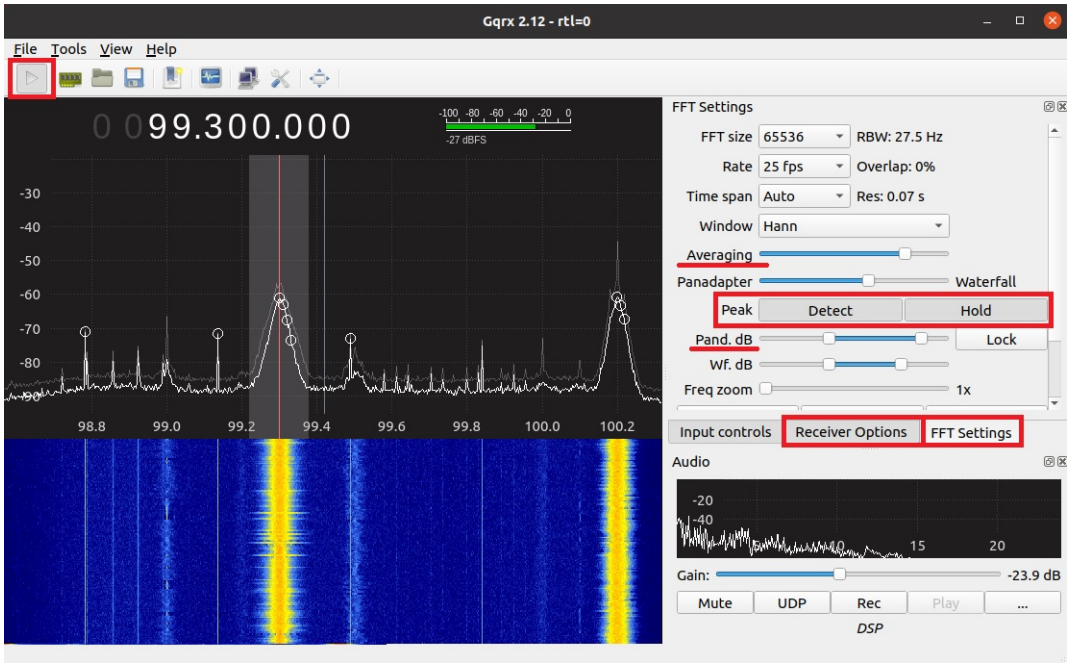
Nyní byl spuštěn program GQRX. V úvodním konfiguračním okně se zvolilo zařízení RTL-SDR (Realtek RTL2838) ze seznamu zařízení (Obr. č. 2.19).

Po kliknutí vlevo nahoře na tlačítko "Start DSP processing" by se měly zobrazit aktivní radio signály ve spektrálním zobrazení. Manuální nastavování frekvence se provádí v záložce "Receiver Options". V záložce "FFT Settings" je možné změnit



Obr. 2.19: Konfigurace zařízení v programu GQRX

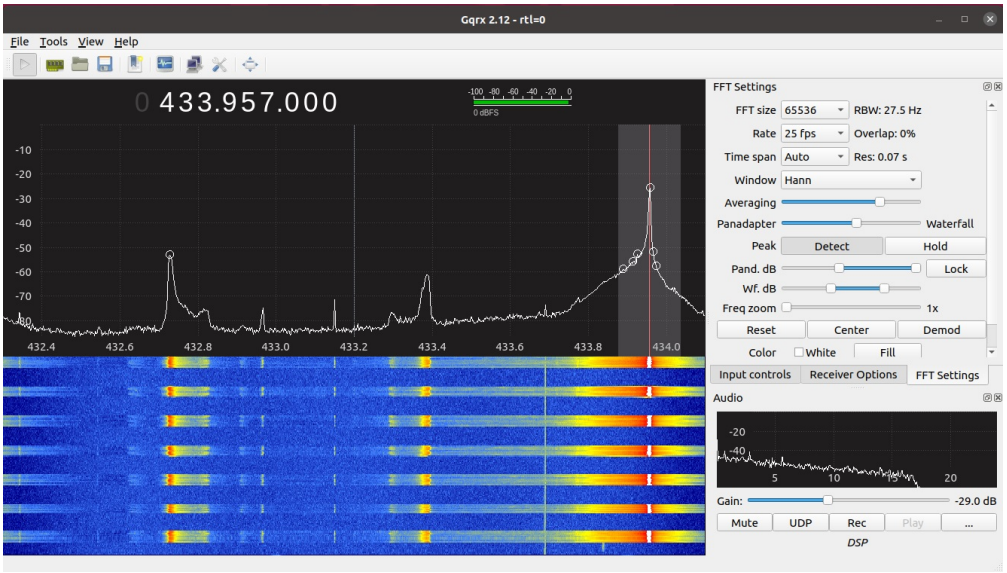
hodnotu "Pandapter dB" pro lepší zobrazení dat. Také hodnotu "Averaging" až téměř k jejímu maximu. Dále pak lze zapnout detekci vrcholů a držet jejich maximálních hodnoty v pozadí. Možností nastavení je zde mnoho. Pro identifikaci signálu a jeho přesné frekvence, který bude zachytáván, však stačí zde uvedené, vyznačené v obrázku č. 2.20.



Obr. 2.20: Prostředí programu GQRX

Arduino nyní bylo připojeno do USB a tím začalo aktivně vysílat data. V programu

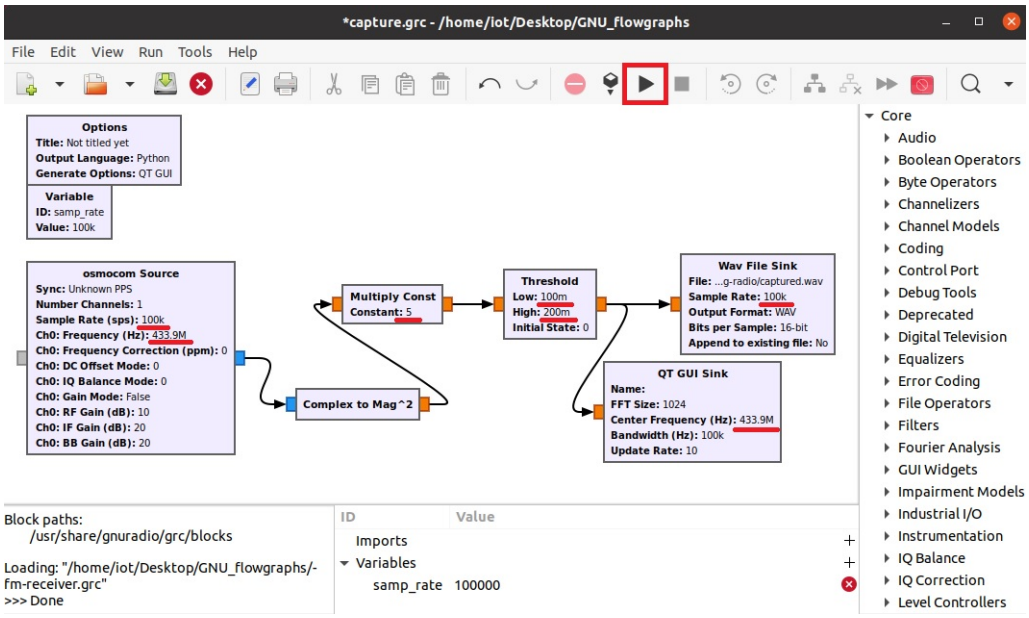
GQRX se změnila centrální frekvence na 433 MHz. Nejvyšší špičky se objevovaly na frekvenci 433,96 MHz, což znamená, že přesná frekvence odesílání dat tohoto zařízení je právě tato (Obr. č. 2.21). Následně se může použít GNURadio na odchycení vysílaného signálu, zpracovat jej a obnovit původní data.



Obr. 2.21: Identifikace frekvence vysílání

2.4.4 Zachycení a zpracování signálů

Spustil se program GNURadio a v něm bylo vytvořeno blokové schéma pro odchycení a zpracování signálu. Jednotlivé bloky je možné vyhledat podle názvu pomocí klávesové zkratky "Ctrl + F". Jako první blok byl přidán "Osmocom Source". Ten zachytává signály ze široké škály zařízení SDR. Nezpracovaný signál obsahuje jak data, tak dodatečné informace týkající se fáze. Pro analýzu je tedy třeba z rádiového signálu odstranit složité části, k čemuž využijeme blok "Complex to Magˆ2". Data vystupující z tohoto bloku jsou typu "float", proto je nutné dodržet tento datový typ i v nastavení následujících bloků. Jakmile je získána datová část signálu, zvýší se její síla "Multiply Const" blokem, který ji násobí nastavenou hodnotou. Následně byl přidán blok "Threshold", který převede aktuální průběh na obdélníkový průběh bez rušivých signálů, kterému lze snadno porozumnět. Další blok "Wav File Sink" zajišťuje ukládání výstupu z "Threshold" bloku. V něm je potřebné si nastavit destinaci uložení získaného souboru. Může se přidat také "QT GUI Sink" blok pro dostání vizuální reprezentace zachytávaného signálu v reálném čase. Aby se zvýšila možnost zachycení datového paketu v plném rozsahu, byla změněna hodnota "sample rate" na 100k. Spojení a vlastnosti všech bloků byly nastaveny viz Obr. č. 2.22.



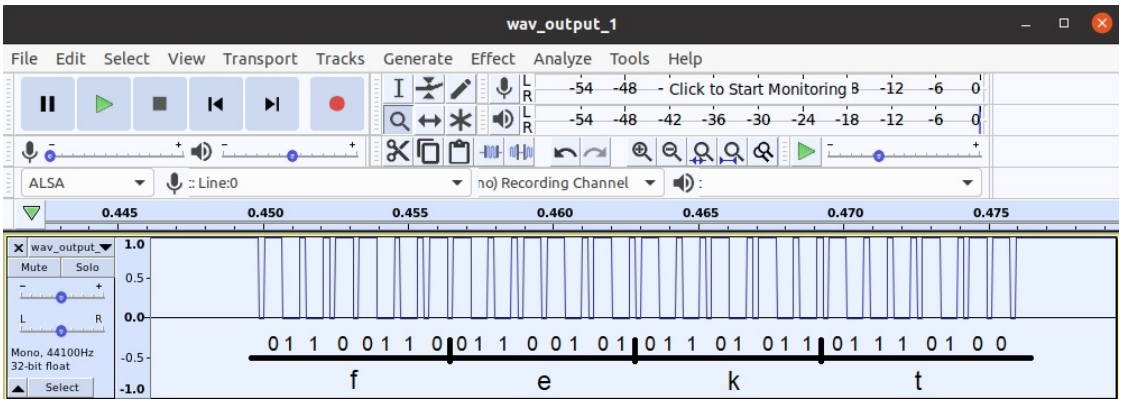
Obr. 2.22: GNURadio graf a nastavení

Po připojení Arduina do USB se spustil pracovní postup v GNURadiu vyznačeným tlačítkem v obrázku č. 2.22. Po chvíli se proces ukončil a tím došlo k zachycení signálu do souboru .wav. Ten pak lze otevřít v nástroji Audacity, ve kterém se zobrazí zachycený průběh, a provede se manuální analýza.

Byl spuštěn program Audacity a přes záložky "File > Import > Raw Data" se vybral získaný soubor .wav. Zobrazil se proud dat opakující paket definovaný v programu Arduina. Byla vybrána libovolná sekvence a přiblížila se pomocí klávesy F4.

Data byla v tomto případě modulována invertovanou On-Off Keying (OOK) technikou (horní hranice = "0"; spodní hranice = "1"), která je jednou z nejběžnějších typů modulace ASK (Amplitude-Shift Keying). OOK je modulační schéma, ve kterém jsou digitální data reprezentována přítomností anebo absencí nosné vlny.

Rozdělí-li se finální data na 8-bitové segmenty a převedou se z binární podoby do ASCII, získají se původní data, která byla odeslána. V tomto případě "fekt" viz obrázek č. 2.23.



Obr. 2.23: Odečtení a převedení signálu do ASCII

2.5 Black Box testování komunikačního prostředí ZigBee

Popis testu

Úkolem tohoto testu je ověření, zda je v komunikaci zakomponované šifrování dat.

Jak provést test manuálně

Během testu ověřte, zda je možné zachycenou komunikaci bez dešifrování zobrazit v původním tvaru.

Testem lze odhalit tyto zranitelnosti

- Nešifrovaná komunikace

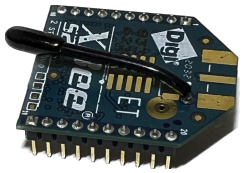
2.6 ZigBee exploitace

V této části je předvedeno, jak vytvořit vlastní zařízení komunikující pomocí protokolu ZigBee, jeho konfigurace a zprovoznění. Následně je pak proveden odchyt komunikace.

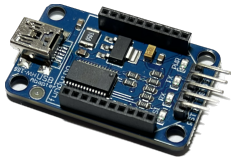
2.6.1 Použité Prostředky

Hardware

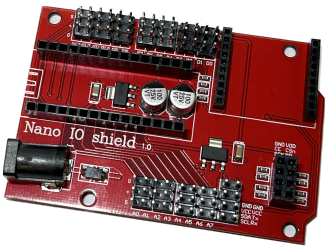
- XBee Modul (Obr. č. 2.24)
- Programátor XBee (Obr. č. 2.25)
- Nano IO Shield 1.0 (Obr. č. 2.26)
- Arduino Nano (Obr. č. 2.27)



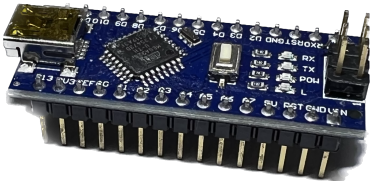
Obr. 2.24: Xbee modul



Obr. 2.25: Programátor XBee

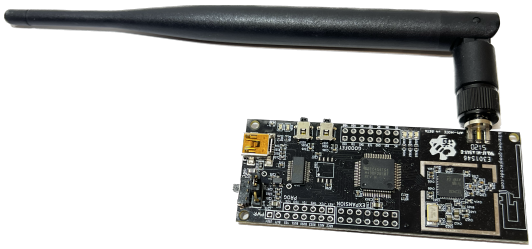


Obr. 2.26: Nano IO Shield 1.0



Obr. 2.27: Arduino Nano

- APIMote (Obr. č. 2.28)



Obr. 2.28: APIMote

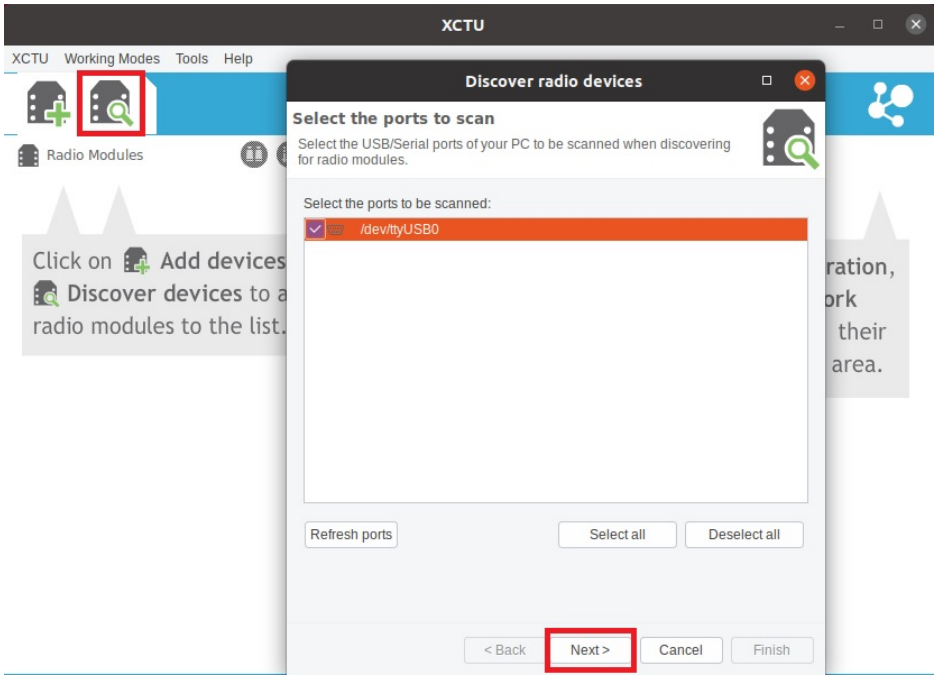
APIMote je open-source hardware určený na zkoumání síťové komunikace podle standardu IEEE 802.15.4, pod který ZigBee spadá.

Software

- Arduino IDE
- XCTU
- KillerBee
- Wireshark

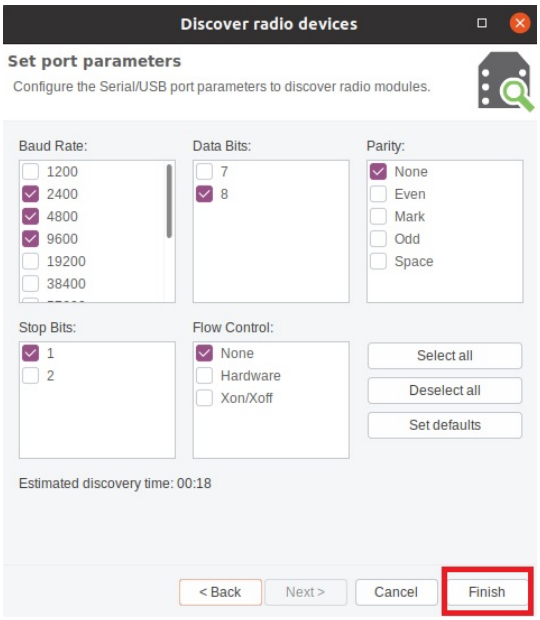
Konfigurace zařízení

V první řadě byla provedena aktualizace firmware XBee Modulu. Pomocí programátoru XBee bylo zařízení připojeno do systému. Po připojení se spustil program XCTU, prostřednictvím kterého se konfigurují XBee moduly.



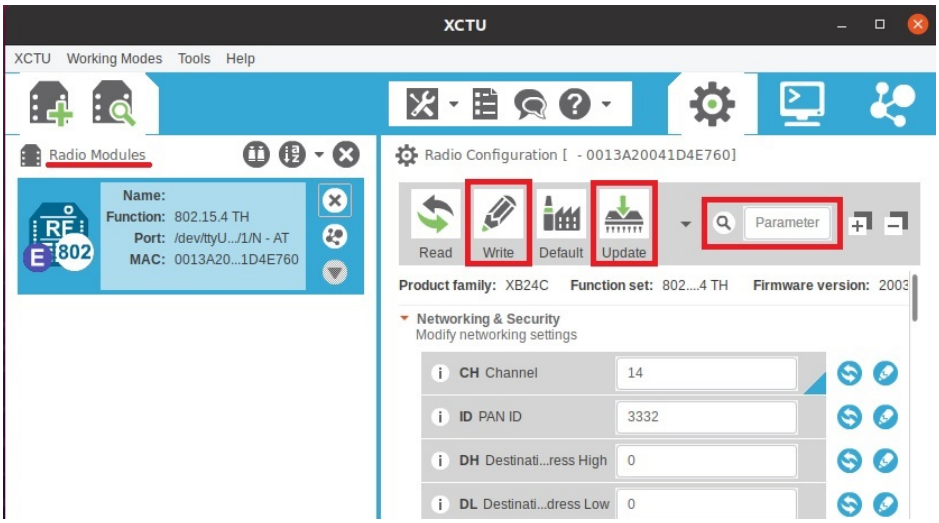
Obr. 2.29: XCTU - Vyhledání zařízení

Kliknutím na tlačítko "Search" zvýrazněného na obrázku č. 2.29 se vyhledávají připojené moduly. Dále se v nabídce vybral odpovídající USB port ke skenování. Poté si program vyžádal nastavení parametrů portu, které bylo provedeno viz obrázek č. 2.30. Kliknutím na tlačítko "Finish" se spustilo vyhledávání. V dalším zobrazeném okně se objevily výsledky s nálezem. Nalezený modul se vybral a tlačítkem "Add selected devices" přidal do programu XCTU.



Obr. 2.30: XCTU - parametry portu

Jakmile bylo zařízení přidáno, kliknutím na zařízení v sekci "Radio Modules" se v pravé části zobrazily všechny jeho konfigurace (Obr. č. 2.31). Pokud tak již není učiněno, je potřebné aktualizovat firmware modulu a to kliknutím na "Update" a zvolením `XB24C > 802.15.4 TH > 2003`. V konfiguračním panelu lze měnit například číslo pracovního kanálu, PAN ID (Personal Area Network Identifier), typ šifrování, rychlost přenosu dat přes rozhraní a podobně. Těmito hodnotami se určuje, jak se XBee modul chová v síti ZigBee.



Obr. 2.31: XCTU - prostředí konfigurace

ZigBee rozděluje pásmo 2,4 GHz na 16 kanálů, od 11 do 26. Číslo kanálu se do konfigurace píše v hexadecimální soustavě. Pro tento případ se zvolil pro modul pracovní kanál číslo 20. Toto číslo je možné zapsat v hexadecimálním tvaru jako 14, takový zápis může mít i podobu 0x14. Program XCTU je schopný pracovat s oběma verzemi tohoto zápisu. Jelikož je seznam parametrů, které lze nastavit velký je vhodné je vyhledávat pomocí textového pole a lupy vyznačeného na obrázku č. 2.31. Pro kanál je to zkratka "CH". Další vlastnost, která se nastavila, byla přenosová rychlost dat na hodnotu 2400 (zkratka pro vyhledání "BD"). Nastavení se nahrálo do modulu tlačítkem "Write", a následně se zařízení odpojilo od systému. Hodnota rychlosti přenášení dat, která byla zadána pro modul Xbee, byla stejná, jaká se použila pro Arduino.

Ve chvíli, kdy bylo Xbee nakonfigurováno, se naprogramovalo Arduino Nano na posílání dat do modulu. Výměna dat mezi Arduinem a modulem Xbee probíhá prostřednictvím sériové komunikace na desce Nano IO Shield, přes kterou jsou porty těchto zařízení propojeny. Byl použit jednoduchý program, který se získal z Google Drive společnosti Attify (viz Obr. č. 2.32).

Obr. 2.32: ArduinoIDE - program pro ZigBee přenos

Program nejprve použije knihovnu SoftwareSerial pro komunikaci přes sériový port na pinech 2 a 3. Pak přes ně odešle daný řetězec "Offensive IoT Exploitation". Posílá se ještě hodnota "a", která je při každém odeslání zprávy inkrementována o 1. Jak bylo již výše zmíněno, hodnota rychlosti přenosu v programu je 2400.

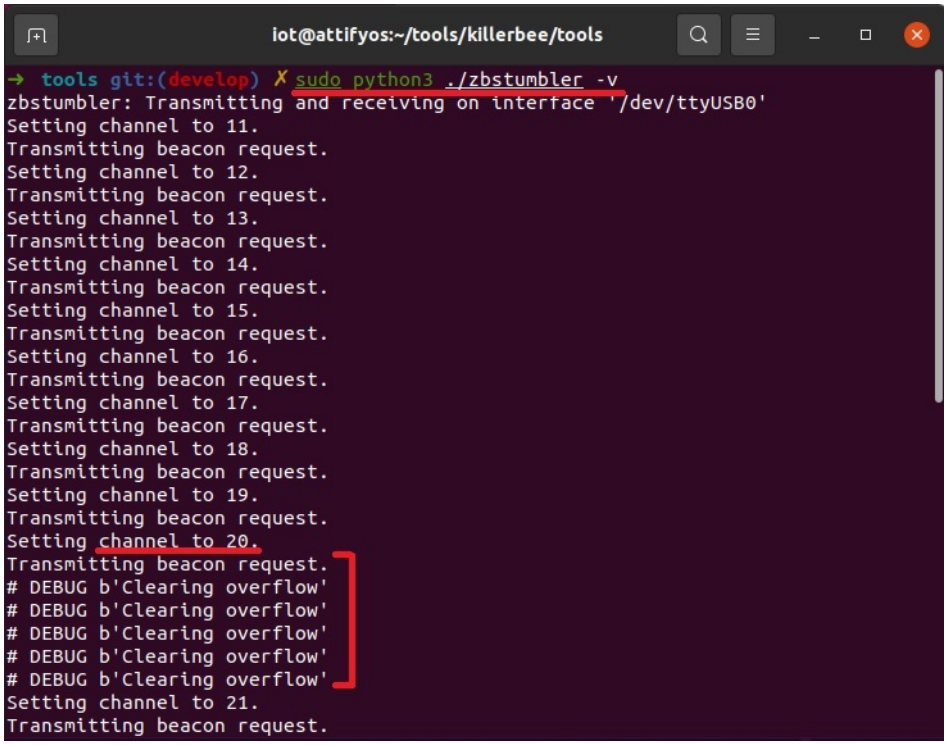
Naprogramování Arduina se provedlo způsobem zmíněným už v předchozí kapitole. Následně se modul Xbee a naprogramované Arduino Nano připojilo na desku Nano IO Shield a celé toto zařízení pak přes Arduino připojilo do systému.

Odchytávání ZigBee paketů

APIMote běží na firmwaru Killerbee, který umožňuje komunikaci se zařízením pomocí nástrojů příkazového řádku. Tento firmware také obsahuje skripty pro útok založený na přehrávání odchycených paketů, tzv. beacon spoofing, útok rušením a další.

Zařízení se připojilo do systému. Kontrola připojení a funkčnosti se provádí spuštěním skriptu `zbid` příkazem `$ sudo python ./zbid`, který vypíše ID zařízení.

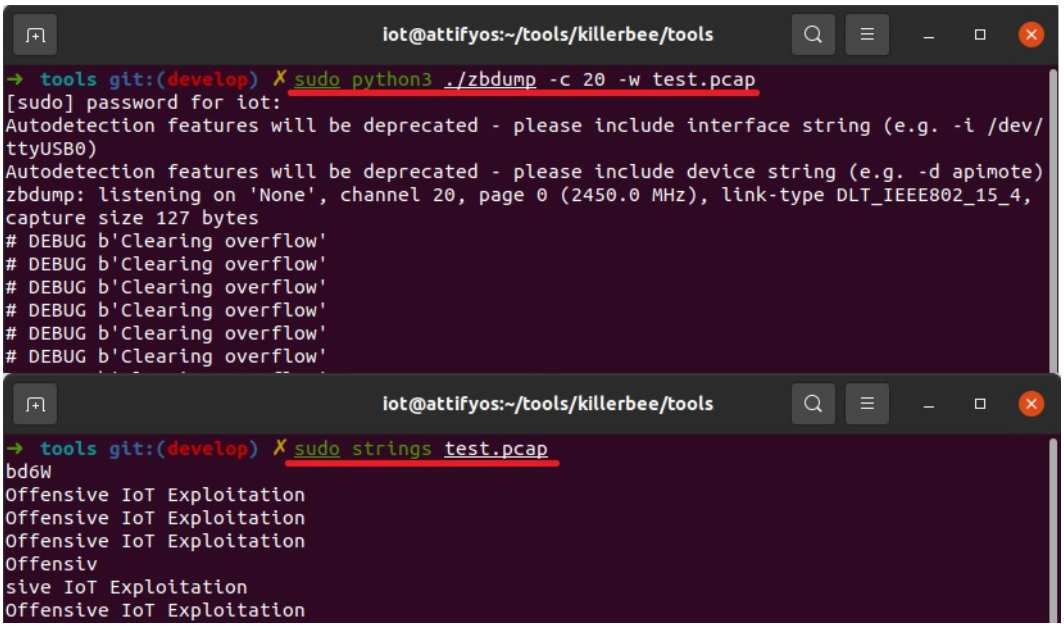
K vyhledání cílového zařízení se provedlo skenování aktivity na jednotlivých kanálech ZigBee. Skriptem `zbstumbler` se prochází jednotlivé kanály a zjišťuje se, zda na některém z nich vysílají nějaká zařízení. V módu rozšířeného výpisu se spustí příkazem `$ sudo python3 ./zbstumbler -v`.



Obr. 2.33: Spuštění skriptu zbstumbler

Z obrázku č. 2.33 je patrné, že na kanále číslo 20 je zjevná aktivita. Bližší prozkoumání dění na kanále 20 se provedlo použitím skriptu `zbdump`, kterým se odchytí probíhající komunikace do souboru. Určení kanálu se provádí značkou `-c` a definování souboru značkou `-w`. Výsledný příkaz se použil ve tvaru: `$ sudo python3 ./zbdump -c 20 -w test.pcap`. Po pár vteřinách, jakmile bylo odchyceno několik paketů se proces ukončil. Ze vzniklého souboru se použitím nástroje `strings` vypsal do konzole jeho obsah k provedení rychlé analýzy. Jak je vidno z obrázku č. 2.34,

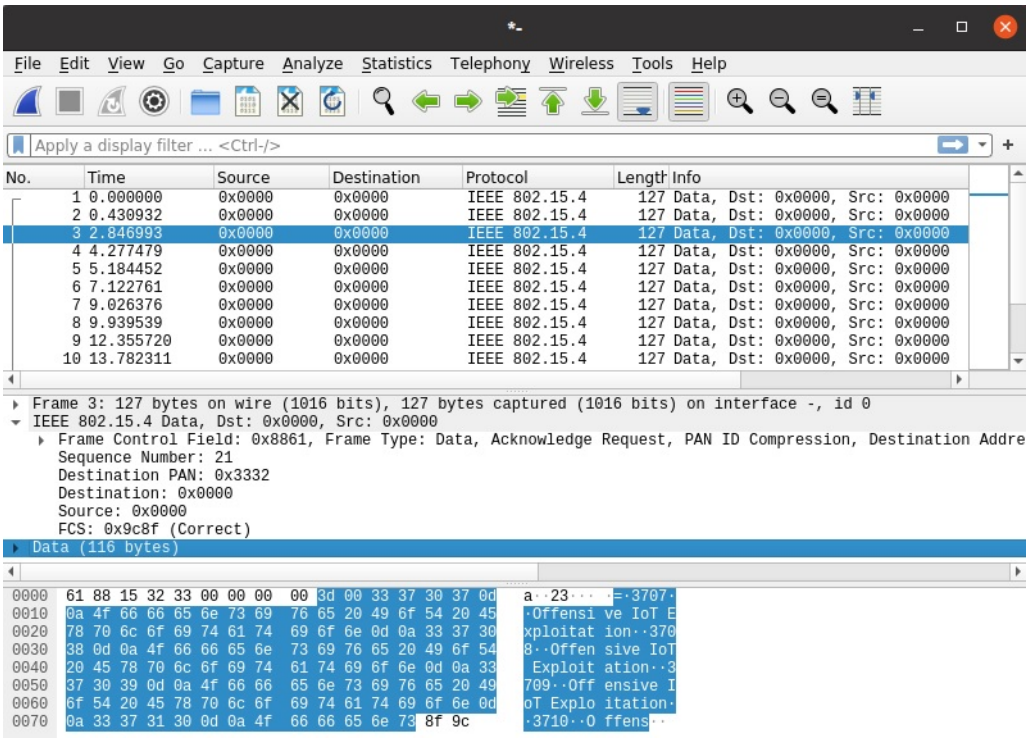
bylo možné úspěšně odchytit pakety, které byly vysílány kombinací zařízení Arduino Nano a Xbee modulem. A je také zřejmé, že při přenosu není použito žádné šifrování.



Obr. 2.34: Odchycení komunikace a její výpis

Pro další analýzu je možné tento soubor otevřít ve Wiresharku. Alternativně je možné použít skript `zbwireshark` namísto `zbdump`. Ten automaticky přenese všechny síťové pakety do programu Wireshark a spustí se příkazem `$ sudo python3 ./zbwireshark -c 20` (Obr. č. 2.35).

U zařízení s malou nebo žádnou ochranou proti přehrávání (replay attack) hrozí, že při odchycení a přehrávání paketů může v určitých případech způsobit převzetí kontroly nad daným zařízením. Modifikováním dat paketů lze provést další útoky na cíle používající Zigbee protokol. Nástroj Killerbee poskytuje spoustu možností útoků, kterými se mohou testovat zařízení na zranitelnosti v zabezpečení.



Obr. 2.35: Odchycení komunikace pomocí zbwireshark

2.7 White Box testování mobilní aplikace

Je v kódu mobilní aplikace pevně ukotvený šifrovací klíč? Je v případě jeho dohledání možné, pomocí analýzy paketů vyměňovaných mezi serverem a zařízením, získat potřebné informace k provedení útoku?

Popis testu

Úkolem tohoto testu je ověření, zda je možné nalézt pevně ukotvený šifrovací klíč v kódu aplikace. V případě jeho nalezení je pak možné analyzovat komunikaci, a získat informace k tomu, jak převzít kontrolu nad zařízením.

Jak provést test manuálně

Dekompilujte soubor ".apk". Pokuste se nalézt knihovnu, ve které by mohly být funkce na šifrování a dešifrování dat. Prozkoumejte tuto knihovnu vhodným nástrojem, který ji analyzuje. Pokuste se najít instrukci zodpovědnou za šifrování a prozkoumejte její vstupní proměnné. V případě získání šifrovacího klíče analyzujte komunikaci mezi serverem a zařízením.

Testem lze odhalit tyto zranitelnosti

- Aplikace používá pevný šifrovací klíč

2.8 Kompromitace mobilní aplikace chytré zásuvky

V této části je popsána analýza mobilní aplikace, prostřednictvím které lze ovládat chytrou zásuvku. Poté je proveden odchyt komunikace. Na základě získaných informací z analýzy aplikace bylo možné provést její dešifrování. Pak se provedl průzkum a srovnání jednotlivých paketů, díky čemuž se získaly informace k provedení útoku.

2.8.1 Použité Prostředky

Hardware

- Chytrá zásuvka Orvibo (Obr. č. 2.36)

Obr. 2.36: Zásuvka Orvibo s redukcí

  Jelikož zásuvka nemá kolíky na český typ zásuvek, bylo potřebné dokoupit redukci.
- Mobilní telefon s aplikací

Software

- HomeMate.apk
  Tento soubor lze získat z Google Play Store.
- JADx
- Ghidra
- Wireshark

2.8.2 Příprava zařízení

Do mobilního telefonu byla stažna aplikace "HomeMate", která je dostupná na Google Play i AppStore. V aplikaci bylo pro používání potřebné provést registraci pomocí emailu a zvolením hesla pro nový účet. Po vytvoření bylo možné spárovat chytrou zásuvku (Orvibo S20C) s aplikací následujícím způsobem.

Zásuvka musí být připojena do sítě. Uvedení chytré zásuvky do konfiguračního módu se provede stisknutím tlačítka, po dobu 6 sekund, umístěného na její přední části. Tlačítko pak začalo červeně blikat a v té chvíli bylo možné ji přes lokální WiFi připojit do sítě. Po dokončení tohoto procesu se dala zásuvka ovládat pomocí mobilní aplikace, tedy zapínat a vypínat.

2.8.3 Analýza mobilní aplikace

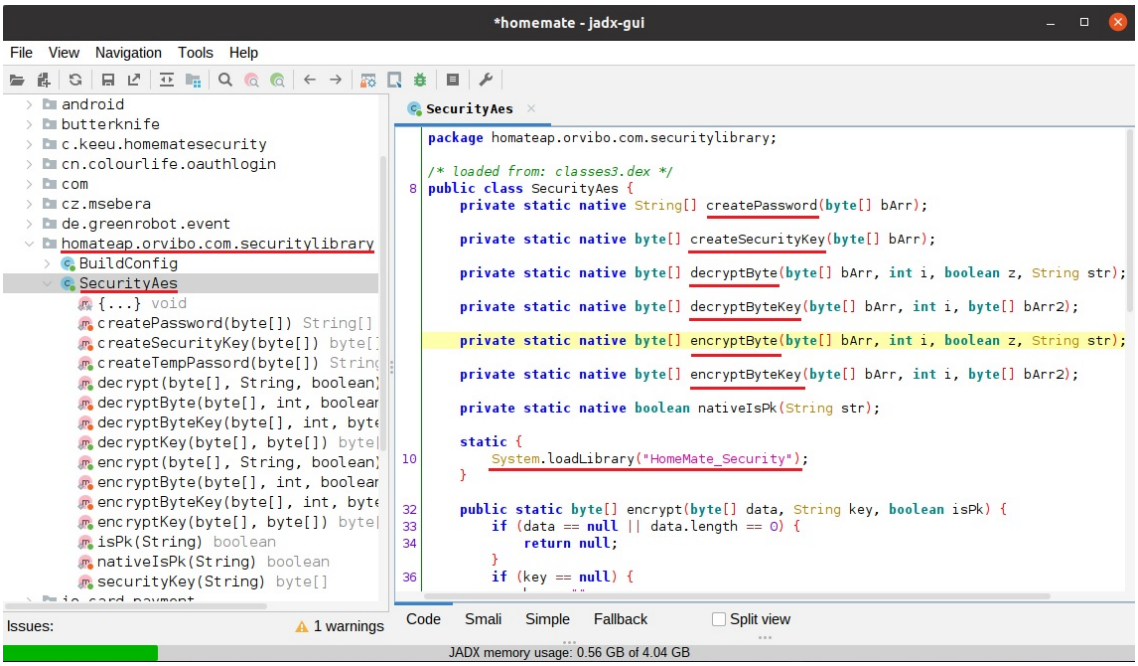
Analyzována byla mobilní aplikace určená pro platformu Android (Homemate.apk). Důvodem, proč byla vybrána aplikace pro Android místo pro iOS je ten, že se lze mnohem snadněji a efektivněji přiblížit k původnímu kódu pomocí dekompilace, než je tomu u aplikací pro iOS. Díky analýze aplikace lze porozumět příkazům zařízení a tomu, jak jsou tyto příkazy do zařízení odesílána. Je možné pak použít tyto informace k manuálnímu vytvoření paketu, jenž umožní kontrolovat stav zařízení.

Aplikace Android jsou archivní balíčky, které obsahují zkompilovaný kód třídy, soubor Android-Manifest.xml, zdroje, nativní knihovny, a další prostředky. K dekompilaci aplikace byl použit nástroj JADx. Tento nástroj je možné spustit v grafickém uživatelském rozhraní (GUI). Příkaz pro spuštění se zadává ve složce "/jadx/build/jadx/bin" ve tvaru: `$ ./jadx-gui`. Pomocí tlačítka "Open File" byl nahrán soubor do JADx a automaticky dekompilován.

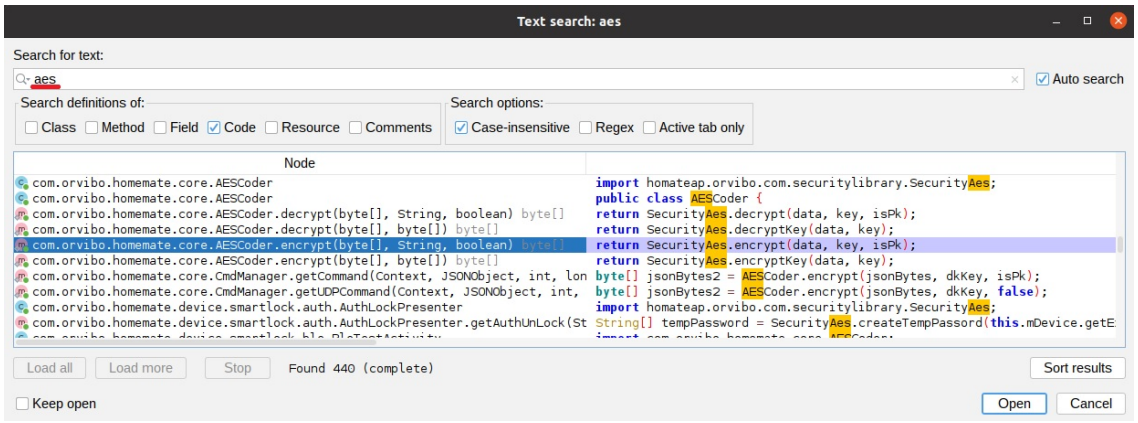
V navigačním panelu v levé části se otevřela třída "SecurityAes" uvnitř "homemateap.orvibo.com.securitylibrary". V této třídě je mnoho zajímavých metod jako například `createPassword`, `decryptByte`, `encryptByte` a další (viz Obr. č. 2.37). Všechny tyto metody jsou obsaženy v nativní knihovně, která se načítá prostřednictvím `System.loadLibrary("HomeMate_Security")`.

Je možné v kódu vyhledat text AES a nalézt další příklady, jak je šifrování v aplikaci řešeno. Textové vyhledávání se nachází v záložce "Navigation > Text search". Bylo nalezeno několik shod, jak ukazuje obrázek č. 2.38.

Pro lepší porozumění byla prozkoumána nativní knihovna, která obsahuje většinu klíčových funkcí pro provádění šifrovacích a dešifrovacích rutin. Nativní knihovna `libHomeMate_Secutrity.so` se nachází v adresáři "/lib", kde jsou architektury `armeabi` a `x86`. Rozbalením souboru "Homemate.apk" byly získány tyto knihovny, provedeno následujícím způsobem.



Obr. 2.37: JADx - výpis třídy SecurityAes

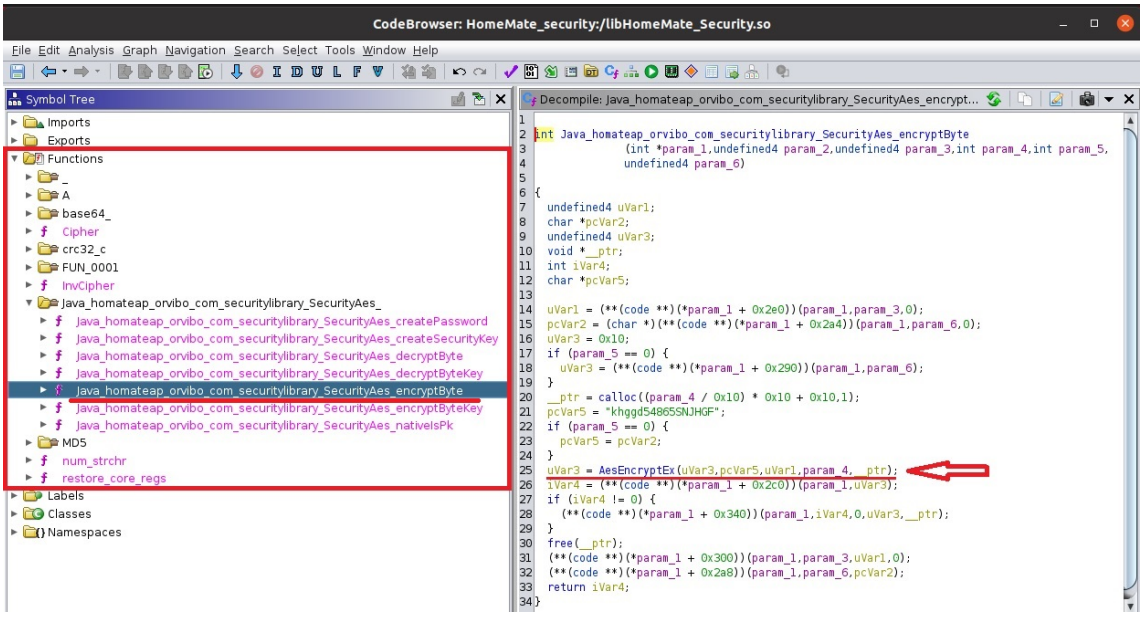


Obr. 2.38: JADx - vyhledání textu v kódu

```
$ unzip Homemate.apk -d homemate
$ cd homemate/lib/armeabi/
$ ls libHomeMate_Security.so
```

Nezáleží na tom, ze které z těchto dvou architektur se zvolí daný soubor. Na prozkoumání tohoto souboru byl použit nástroj Ghidra. Jeho spuštění se provádí příkazem `$ ./ghidraRun`. Byl vytvořen nový projekt a importoval se soubor `libHomeMate_Security.so`. Po načtení se dvojím kliknutím na soubor spustila

analýza. V okně zobrazeném v levé části programu je seznam funkcí přítomných v knihovně (Obr. č. 2.39).



Obr. 2.39: Ghidra - výpis funkcí v knihovně

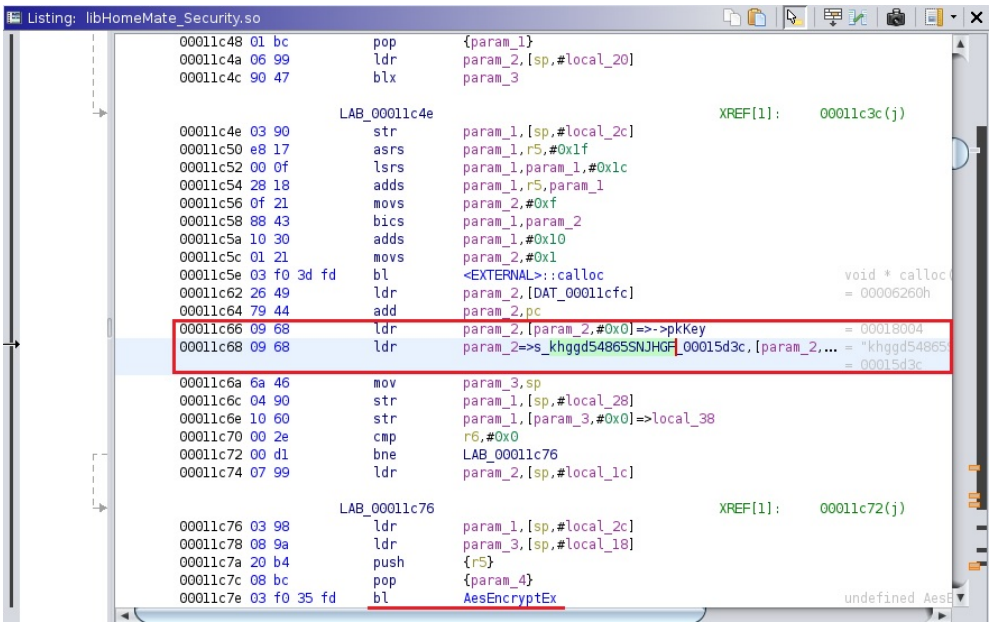
Funkce začínající na "Java_homateap_orvibo" jsou implementací nativních funkcí z kódu jazyku Java. Byla analyzována funkce `encryptByte` zvýrazněna v obrázku č. 2.39. V jejím zobrazení po dekompilaci je jako poslední instrukce `AesEncryptEx`. Kliknutím na ni dojde v okně "Listing" k přesunutí na její pozici ve výpisu knihovny `libHomeMate_Security.so`.

Instrukce `bl` v architektuře ARM znamená skok na adresu v ní definovanou a uložení návratové adresy do registru odkazů. Ve zobrazení dekompilovaného kódu, je zřejmé, že funkce `AesEncryptionEx` je volána s pěti argumenty (řádek 25). V ARM jsou argumenty ukládány do registrů r0 až r3 a případné další jsou ukládány do zásobníku.

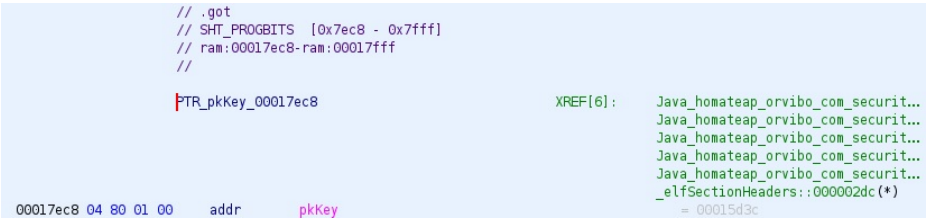
Na adrese 11c66 jsou pod sebou dvě instrukce vyznačené na obrázku č. 2.40. Z nich vyplývá, že druhý argument ukazuje na pkKey, jehož hodnota je zobrazena i na dalším řádku. Alternativně bylo možné poklepat na pkKey pro přesunutí na jeho umístění (Obr. č. 2.41).

Tím se přesunulo k položce pkKey v tabulce Global Offset Table (GOT), která obsahuje skutečnou adresu pkKey. Poklepáním na ni opět došlo k přesunutí, a to už právě na adresu řetězce (Obr. č. 2.42).

Hodnota pkKey je "khggd54865SNJHGF". Toto je klíč AES, který aplikace používá v ECB módu k šifrování a dešifrování.



Obr. 2.40: Ghidra - výpis knihovny



Obr. 2.41: Ghidra - adresa pkKey

Obr. 2.42: Ghidra - hodnota pkKey

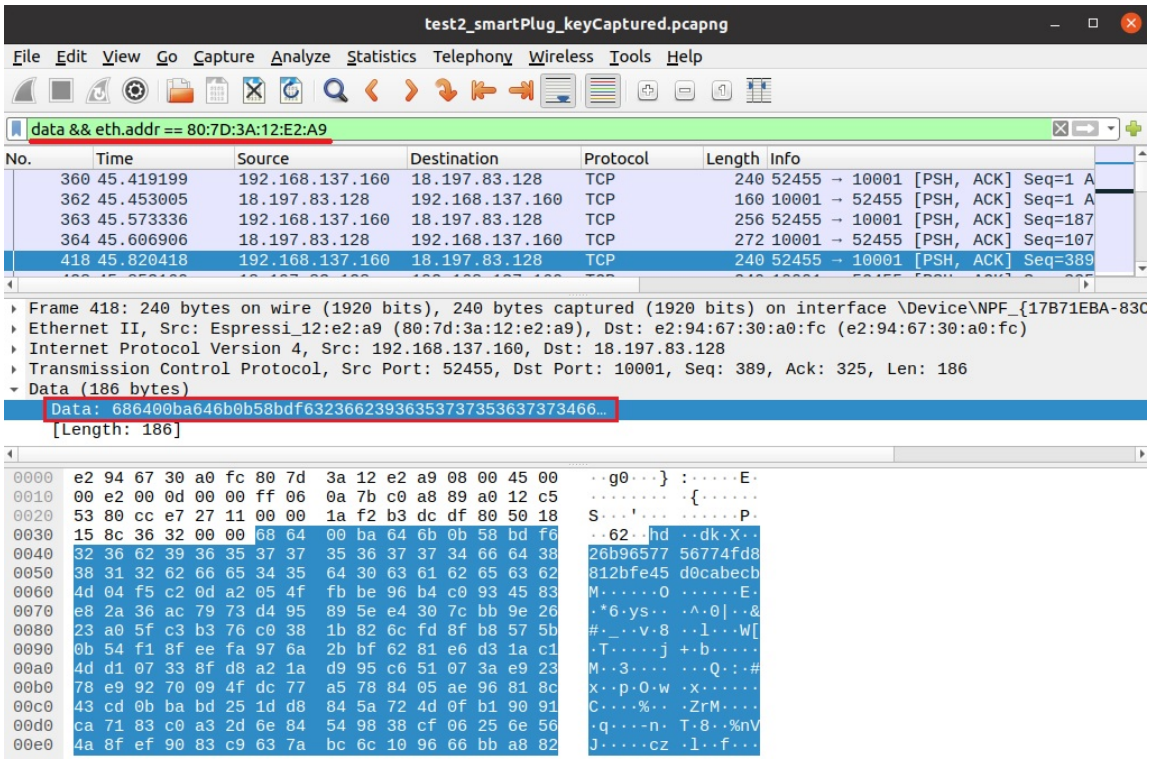
2.8.4 Odchyt síťového provozu

Aby bylo možné provést analýzu síťového provozu chytré zásuvky, bylo potřebné v hostitelském systému (ne ve virtuálním stroji) nastavit hotspot. Poté nakonfigurovat zásuvku tak, aby se připojovala k tomuto bodu, který zároveň poskytuje přístup k internetu.

Pomocí aplikace HomeMate bylo tedy zařízení připojeno k tomuto přístupovému

bodu stejným způsobem jak je uvedeno v kapitole Příprava zařízení. Spustil se Wireshark na rozhraní, kde byl nastavený hotspot. Během zachycování sítového provozu bylo pomocí mobilní aplikace odesláno několik on/off příkazů chytré zásuvce. Pak se odchyt zastavil.

Aby se eliminovala nerelevantní síťová komunikace, byl ve Wiresharku použit filtr `data && eth.addr==80:7D:3A:12:E2:A9`, kde "80:7D:3A:12:E2:A9" je MAC adresa chytré zásuvky (Obr. č. 2.43).



Obr. 2.43: Wireshark - filtr odchycené komunikace

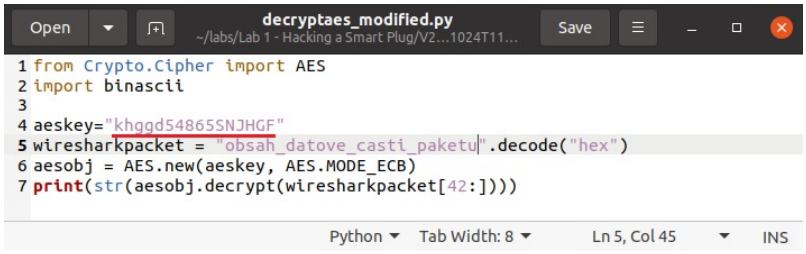
IP adresa 18.197.83.128 patří homemate.orvibo.com, což lze ověřit díky příkazu `dig`, který dotazuje DNS servery.

```
$ dig +short homemate.orvibo.com
access-web-prd-de2-80189dd559508282.elb.eu-central-1.amaz
onaws.com.
18.197.83.128
```

Z odchycené komunikace ve Wiresharku bylo potvrzeno že je obsah paketů zašifrovaný. Po analýze několika paketů byla určena jejich obecná struktura. Každý paket obsahuje následující součásti:

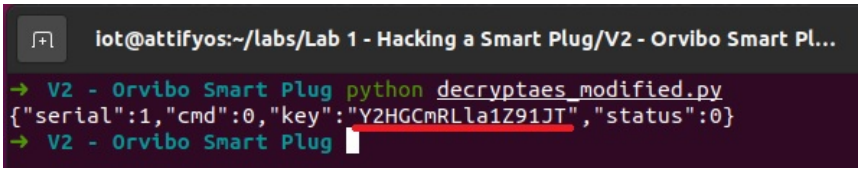
- 2 bajty - Označení (hd)
- 2 bajty - Velikost celého paketu
- 2 bajty - Typ paketu (pk/dk)
- 4 bajty - CRC32 kontrolní součet obsahu paketu
- 32 bajtů - ID relace
- proměnná délka - Zašifrovaná data

Jelikož už byly zjištěny podrobnosti o šifrování, je možné vytvořit skript pro dešifrování dat v paketu. Byl použit jednoduchý skript od společnosti Attify v podobě na obrázku č. 2.44. Skript se spouští příkazem `$ python decryptaes_modified.py`.



Obr. 2.44: Skript na dešifrování AES

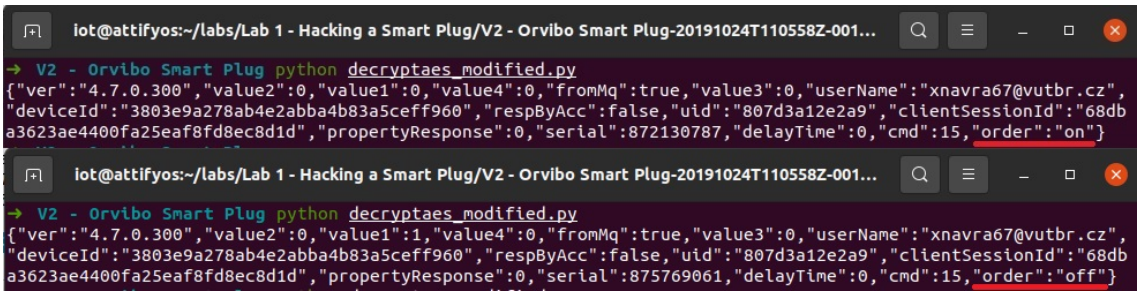
Zašifrovaná vstupní data do skriptu byla vkládána v hexadecimálních hodnotách. Tato hodnota se z paketu získala pravým kliknutím myši na datový řetězec (označený na obrázku č. 2.43) a výběrem "Copy > ...as a Hex Stream". A vkládá se do proměnné `wiresharkpacket`.



Obr. 2.45: Nový šifrovací klíč

Spuštěním skriptu se získá původní obsah paketu. Na obrázku č. 2.45 je zobrazeno dešifrování prvního paketu, ve kterém chytrá zásuvka obdrží od webového serveru nový šifrovací klíč. Tento klíč je pak používán k šifrování všech následujících paketů. Proto byl tento klíč zkopírován a ve skriptu aktualizována hodnota `aeskey`. Nyní mohly být dešifrovány pozdější pakety (Obr. č. 2.46).

Bylo tedy možné dešifrování celé komunikace mezi chytrou zásuvkou a koncovým bodem na internetu. Díky tomu se mohla provést analýza paketů. K jednoznačnému identifikování relace, odesílá server v hlavičce paketu také jedinečný 32 bajtový

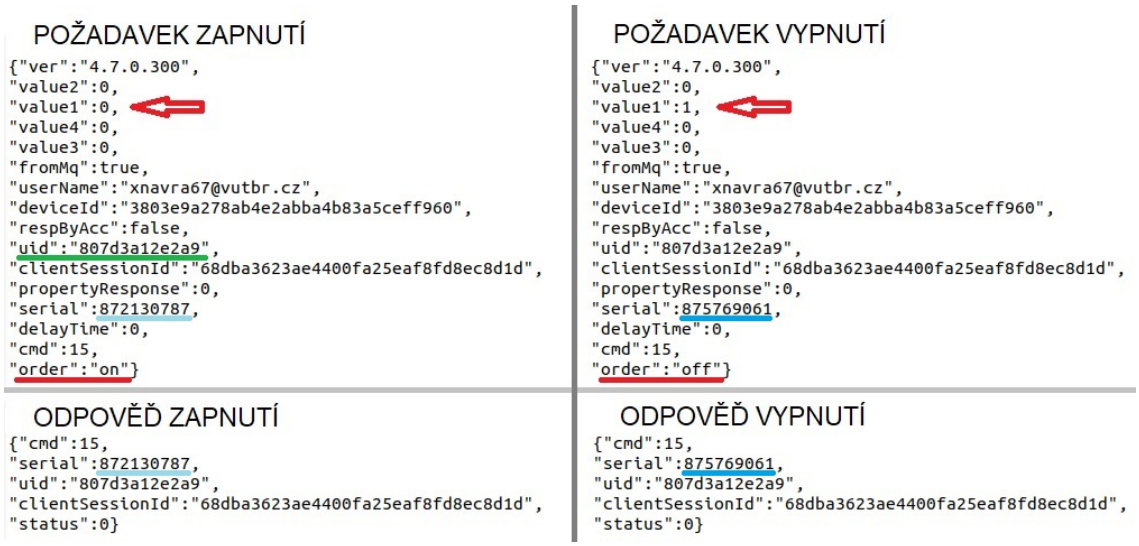


Obr. 2.46: Dešifrované pakety ON/OFF

řetězec (ID relace). Na pakety se stejným ID relace je k šifrování používán stejný klíč AES.

2.8.5 Analýza dat paketů

Zprávy jsou odesílány chytré zásuvce ze serveru homemate, nikoliv z aplikace, což už vyplynulo z IP adres zdroje a cíle ve Wiresharku. Zásuvka tedy nepracuje v režimu peer-to-peer.



Obr. 2.47: Porovnání požadavků a odpovědí

Zprávy jsou psány ve stylu JSON standardu. Z obrázku Porovnání požadavků a odpovědí (Obr. č. 2.47), pro zapnutí a vypnutí vyplývá nasledující:

- všechny zprávy jsou označeny sériovým číslem, které se zvyšuje o 1
- požadavek a odpověď na něj mají stejné sériové číslo
- "value1":0 odpovídá zapnutému stavu zásuvky
- "value1":1 odpovídá vypnutému stavu zásuvky
- `uid` je stejné jako MAC adresa zásuvky

Na základě výše uvedených informací je možné napsat skript pro ovládání chytré zásuvky, ovšem už komplikovanější než předchozí skripty. Protože zásuvka reaguje pouze na příkazy ze serveru homemate, bylo by potřebné vytvořit proxy server. Ten by měl dokázat předávat nejen pakety mezi zásuvkou a serverem, ale mít také možnost upravovat zprávy. Musí být zajištěno, aby pozměněná zpráva byla zašifrována způsobem, který chytrá zásuvka očekává. Zásuvka odpovídá jak na příkazy přijaté ze serveru, tak na příkazy upravené útočníkem. Odpovědi, které jsou reakcí na upravené pakety, musí být zahozeny, aby se nedostaly na server homemate. Všechny ostatní požadavky a odpovědi mezi zásuvkou a serverem musí být transparentně předávány.

Kombinací výše uvedené techniky a útoku známého jako "Útok s falešným dvojčetem" (Evil Twin attack) může útočník převzít kontrolu nad jakoukoliv chytrou zásuvkou, fungující na stejném principu, bez vědomí oběti. Taková zásuvka se ochotně pokusí připojit k jakékoliv WiFi síti, která má stejný název (SSID) jako ta, s níž byla spárována. Nezáleží na tom, zda byla zásuvka původně spárována se zabezpečenou sítí WPA2. Útočník může vytvořit falešný přístupový bod se stejným SSID bez zabezpečení. Odesláním cílených deautentizačních paketů (Wi-Fi deauthentication attack) může útočník vyřadit zásuvku ze zabezpečené sítě a ta se pak pokouší dostat zpět do sítě. Díky silnějšímu signálu by se mohla připojit právě k přístupovému bodu útočníka.

Závěr

Tuto práci jsem si zvolil proto, že jsem vnímal bezpečnost IoT jako zajímavé téma. V průběhu práce se mi tato moje domněnka potvrdila a díky ní jsem pochopil, jak je situace se zabezpečením IoT produktů vážná.

Výrobci se při návrhu téměř nevěnují budoucí bezpečnosti zařízení, která vyrábí, jelikož by jim to zvedlo náklady a tím i výslednou cenu. Poté by mohlo dojít k propadu prodaných kusů. Zatím výrobcům žádná legislativa nezakazuje takto zranitelná zařízení prodávat. Jelikož jdou stále na odbyt, nic je nemotivuje kvalitu z bezpečnostního hlediska zlepšovat. S tím, jak se počet IoT zařízení ve světě rychle zvětšuje, stahuje na sebe pozornost mnoha hackerů. Zabezpečení IoT sítí je ve většině případů chabá, protože si ji snaží vytvořit sami koncoví zákazníci, aniž by této problematice dostatečně rozuměli.

V teoretické části byla rozebrána problematika IoT, její současný stav a kyberkriminalita s ní spojená. Jak již bylo zmíněno výše, současný stav zabezpečení ve většině domácností je nevyhovující. Dále byly rozebrány vybrané komunikační protokoly, jako například ZigBee a BLE vyplývající ze standardu IEEE 802.15.4. V této části práce jsme se dotkli i popisu BlackBox a WhiteBox testování. Společnost OWASP vytvořila standard ISVS a jeho jednotlivé části byly také popsány v teoretické rovině. Poslední částí jsou komplexní kontrolní seznamy vztahující se k problematice v této práci.

Druhá rozsáhlá část práce se týkala samotného praktického prezentování zranitelností vybraných IoT zařízení. Penetračním testováním byly ověřovány zranitelnosti jako nešifrovaná komunikace, nevyžadování autentizace, možné ovládaní zařízení neoprávněnou osobou a používaní pevného šifrovacího klíče aplikací. Praktická část je rozdělena do čtyř pomyslných částí, přičemž každá z nich se věnuje jinému IoT zařízení. Každá táto část je rozdělena na dvě, a to popis BlackBox případně WhiteBox testovaní a samotný popis manuálního testovaní.

Táto práce může byt přiblížením k problematice zabezpečení IoT zařízení a předvedení manuálního testování zranitelností. Může také sloužit jako učební materiál pro vhled do aktuálního stavu.

Literatura

[1] PATEL, Keyur K; Sunil M PATEL International Journal of Engineering Science and Computing *Internet of Things-IOT: Definition, Characteristics, Architecture, Enabling Technologies, Application & Future Challenges.* [online] Vadodara, Gujarat, India, Květen 2016 [cit. 2021-10-29]

[2] Vanito Hoang *Co je základní kámen IoT - A co to znamená pro čtvrtou průmyslovou revoluci?* [online] Dostupné z URL:<`https://www.thegioimaychu.vn/blog/tong-hop/nen-tang-iot-la-gi-va-y-nghia-cua-no/trong-cuoc-cach-mang-cong-nghe-lan-thu-4-p963/`>. [cit. 2021-10-29]

[3] FAROOQ, M.U.; Muhammad WASEEM; Sadia MAZHAR; Anjum KHAIRI; Talha KAMAL *A Review on Internet of Things (IoT).* 2015 International Journal of Computer Applications. 113. 1-7. 10.5120/19787-1571. [cit. 2021-10-30]. ISSN 0975 8887.

[4] PONTIN, Jason *ETC: Bill Joy's Six Webs.*[online] Září 2005 [cit. 2021-10-31] Dostupné z URL:
<`https://www.technologyreview.com/2005/09/29/230292/etc-bill-joys-six-webs/`>.

[5] ASHTON, Kevin *That 'Internet of Things' Thing. RFID Journal.* [online] Červen 2009 Dostupné z URL:
<`https://www.itrco.jp/libraries/RFIDjournal-That%20Internet%20of%20Things%20Thing.pdf`>. [cit. 2021-10-31]

[6] JAY, Allan. *Number of Internet of Things (IoT) Connected Devices Worldwide 2022/2023: Breakdowns, Growth & Predictions.* Financesonline.com [online]. 2023. Dostupné z URL:<`https://financesonline.com/number-of-internet-of-things-connected-devices/`>. [cit. 2023-04-10]

[7] HOWARTH, Josh. *80+ Amazing IoT Statistics (2023-2030).* Explodingtopics.com [online]. 2023. Dostupné z URL:
<`https://explodingtopics.com/blog/iot-stats`>. [cit. 2023-04-10]

[8] SINHA, Satyajit *State of IoT 2021: Number of connected IoT devices growing 9% to 12.3 billion globally, cellular IoT now surpassing 2 billion.* [online] Září 2021. Dostupné z URL:
<`https://iot-analytics.com/number-connected-iot-devices/`>. [cit. 2021-11-07]

[9] NESHENKO, Nataliia; Bou-Harb, Elias; Crichigno, Jorge; Kaddoum, Georges; Ghani, Nasir *An exhaustive survey on IoT vulnerabilities and a first empirical look on Internet-scale IoT exploitations.* IEEE Communications Surveys & Tutorials, Duben 2019, 21.3: 2702-2733 [cit. 2021-11-12]

[10] SEREDA, Bohdana. *SUPPORTING END USERS IN SECURING IOT-ENABLED SMART HOME DEVICES.* [online] Ottawa, Ontario, Canada, 2022 Diplomová práce. Carleton University. Dostupné z URL: <https://curve.carleton.ca/515027b9-b345-4f8a-afe2-638ca271d83a>. [cit. 2023-04-15]

[11] Odbor bezpečnostní politiky *Bezpečbečnostní hrozby.* [online] 2019 Dostupné z URL: <https://www.mvcr.cz/clanek/bezpecnostni-hrozby-337414.aspx?q=Y2hudW09Mw%3D%3D>. [cit. 2023-03-01]

[12] WETSMAN, Nicole. *Woman dies during a ransomware attack on a German hospital.* [online] 2020 Dostupné z URL: <https://www.theverge.com/2020/9/17/21443851/death-ransomware-attack-hospital-germany-cybersecurity?fbclid=IwAR2SbdZHAxg8sM3OjYUG2GLtKblgJ4sZwDXK6hdakgxL4J60L1aP7saFZC4>. [cit. 2023-04-15]

[13] BUXTON, Oliver. *What Is the Mirai Botnet?.* [online] 2022 Dostupné z URL: <https://www.avast.com/c-mirai>. [cit. 2023-04-14]

[14] DRAGOMIR, Dan, et al. *A survey on secure communication protocols for IoT systems.* International Workshop on Secure Internet of Things (SIoT). IEEE, 2016. p. 47-62. [cit. 2021-11-17]

[15] AL-SARAWI, Shadi, et al. *Internet of Things (IoT) communication protocols.* 2017 8th International conference on information technology (ICIT). IEEE, 2017. p. 685-690. [cit. 2021-11-16]

[16] OLSSON, Johnas *6LoWPAN demystified.* Texas Instruments [online]. 2014 [cit. 2021-11-16] Dostupné z: https://www.ti.com/lit/wp/swry013/swry013.pdf

[17] LI, Hongwei; JIA, Zhongning; XUE, Xiaofeng *Application and analysis of ZigBee security services specification.* 2010 Second International Conference on Networks Security, Wireless Communications and Trusted Computing. IEEE, 2010. p. 494-497. [cit. 2021-11-20]

[18] RAMYA, C. Muthu; SHANMUGARAJ, M.; PRABAKARAN, R. *Study on ZigBee technology.* 2011 3rd International Conference on Electronics Computer Technology. IEEE, 2011. p. 297-301. [cit. 2021-11-20]

[19] MIKHAYLOV, Konstantin; PLEVRITAKIS, Nikolaos; TERVONEN, Jouni. *Performance analysis and comparison of Bluetooth Low Energy with IEEE 802.15. 4 and SimpliciTI.* Journal of Sensor and Actuator Networks, 2013, 2.3: 589-613. [cit. 2021-11-22]

[20] *BLE Protocol Stack — Controller.* In: Pcng.medium.com [online]. 2019 Dostupné z URL:<`https://pcng.medium.com/ble-protocol-stack-controller-2d2d5371deec`>. [cit. 2021-11-22]

[21] GRAZIANO, Mariagrazia; CATANZARO, Daniela. *Study and investigation of Bluetooth Low Energy security in the IoT environment.* 2020. [cit. 2021-11-27]

[22] GOMEZ, Carles; OLLER, Joaquim; PARADELLS, Josep. *Overview and evaluation of bluetooth low energy: An emerging low-power wireless technology.* Sensors, 2012, 12.9: 11734-11753. [cit. 2021-11-27]

[23] SHARMILA, D.; NEELAVENI, R.; KIRUBA, K. *Notice of Violation of IEEE Publication Principles: Bluetooth Man-In-The-Middle attack based on Secure Simple Pairing using Out Of Band association model.* International Conference on Control, Automation, Communication and Energy Conservation. IEEE, 2009. p. 1-6. [cit. 2021-11-27]

[24] MADUGULA, Sai Swaroop; WEI, Ruizhong. *An enhanced passkey entry protocol for secure simple pairing in bluetooth.* arXiv preprint arXiv:2101.09381, 2021. [cit. 2021-11-27]

[25] JOHARI, Rahul, et al. *Penetration Testing in IoT Network.* In: 2020 5th International Conference on Computing, Communication and Security (ICCCS). IEEE, 2020. p. 1-7. [cit. 2021-12-07]

[26] YADAV, Geeta, et al. *Penetration Testing Framework for IoT.* In: 2019 8th International Congress on Advanced Applied Informatics (IIAI-AAI). IEEE, 2019. p. 477-482. [cit. 2021-12-07]

[27] MAHAK, Jain. *Differences between Black Box Testing vs White Box Testing.* Geeksforgeeks.org [online] 2023 Dostupné z URL:<`https://www.geeksforgeeks.org/differences-between-black-box-testing-vs-white-box-testing/`>. [cit. 2023-04-19]

[28] BASSEM, Cédric; GUZMAN, Aron; et al. *OWASP IoT Security Verification Standard.*[online] Version 1.0, 22 January 2021 Dostupné z URL:<https://github.com/OWASP/IoT-Security-Verification-Standard-ISVS>. [cit. 2021-12-08]

[29] AL ATTAR, Rama, et al. *Risk assessment for emerging domains (IOT, cloud computing, and AI).*[online] 2021 IEEE Jordan International Joint Conference on Electrical Engineering and Information Technology (JEEIT). IEEE, 2021. p. 120-127. Dostupné z URL:
<https://ieeexplore.ieee.org/abstract/document/9634156>. [cit. 2023-04-16]

[30] SHIRGUR, Pranav; CHAURASIA, Sandeep. *Development of Secure IoT Ecosystems for Healthcare.*[online] 3rd International Conference on Intelligent Communication and Computational Techniques (ICCT). IEEE, 2023. p. 1-6. Dostupné z URL:
<https://ieeexplore.ieee.org/abstract/document/10076119>. [cit. 2023-04-16]

[31] FARHA, Fadi; CHEN, Hongsong. *Mitigating replay attacks with ZigBee solutions.* Network Security, 2018, 2018.1: 13-19. [cit. 2021-12-11]

[32] OLAWUMI, Olayemi, et al. *Three practical attacks against ZigBee security: Attack scenario definitions, practical experiments, countermeasures, and lessons learned.* In: 2014 14th International Conference on Hybrid Intelligent Systems. IEEE, 2014. p. 199-206. [cit. 2021-12-11]

[33] UKE, S. N.; MAHAJAN, A. R.; THOOL, R. C. *UML modeling of physical and data link layer security attacks in WSN.* International Journal of Computer Applications, 2013, 70.11. [cit. 2021-12-11]

[34] DEOGIRIKAR, Jyoti; VIDHATE, Amarsinh. *Security attacks in IoT: A survey.* In: 2017 International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud)(I-SMAC). IEEE, 2017. p. 32-37. [cit. 2021-12-11]

[35] JARIWALA, Chirag. *OWASP IoT Top 10 Series: Weak or Hardcoded Password Policy OWASP.* [online] February 15, 2021 Dostupné z URL:<https://blog.securelayer7.net/owasp-iot-top-10-series-weak-or-hardcoded-password-policy-owasp/>. [cit. 2021-12-12]

[36] BOEHM, Ellen. *Top 10 IoT Vulnerabilities in Your Devices.* [online] October 28, 2020. Dostupné z URL:<https://www.keyfactor.com/blog/top-10-iot-vulnerabilities-in-your-devices/#weak-guessable-passwords-4>. [cit. 2021-12-12]

Seznam symbolů a zkratek

IoT	Internet věcí - Internet of Things
IEEE	Institut pro elektrotechnické a elektronické inženýrství - Institute of Electrical and Electronics Engineers
ISVS	Standard ověřování zabezpečení internetu věcí - IoT Security Verification Standard
BLE	Nízkoenergetická technologie Bluetooth - Bluetooth Low Energy
RTL-SDR	Softwarově definované rádio v reálném čase - Real-Time Software Defined Radio
DDoS	Distribuované odepření služby - Distributed Denial of Service
AES	Pokročilý standard šifrování - Advanced Encryption Standard
CCM	Čítač s šifrovým blokovým řetězením kódu pro ověřování zpráv - Counter with Cipher block chaining Message authentication code
6LoWPAN	IPv6 v bezdrátových osobních sítích s nízkou spotřebou energie - IPv6 over Low-Power Wireless Personal Area Networks
SKKE	Zřízení symetrického klíče - Symmetric-Key Key Establishment
PHY	Fyzická vrstva - Physical layer
LL	Linková vrstva - Link Layer
L2CAP	Protokol řízení a přizpůsobení logického spoje - Logical Link Control and Adaptation Protocol
QoS	Kvalita služeb - Quality of Services
GAP	Generický přístupový profil - Generic Access Profile
SMP	Bezpečnostní manažer - Security Manager
ATT	Protokol atributů - Attribute Protocol
GATT	Obecný profil atributů - Generic Attribute Profile
HCI	Hostitelský ovladač Rozhraní - Host Controller Interface
MIC	Kontrola integrity zpráv - Message Integrity Check

CMAC	Kód pro ověřování zpráv založený na šifrách - Cipher-based Message Authentication Code
LTK	Dlouhodobý klíč - Long Term Key
CSRK	Řešící klíč podpisu připojení - Connection Signature Resolving Key
IRK	Klíč k řešení identity - Identity Resolving Key
TK	Dočasný klíč - Temporary Key
ECDH	Diffie-Hellmanova eliptická křivka - Elliptic-curve Diffie–Hellman
OWASP	Projekt zabezpečení otevřených webových aplikací - Open Web Application Security Project
API	Rozhraní pro programování aplikací - Application Programming Interface
TLS	Zabezpečení transportní vrstvy - Transport Layer Security
OTP	Jednorázově programovatelné - One-Time Programmable
MMU	Jednotka správy paměti - Memory Management Unit
MQTT	Message Queuing Telemetry Transport
SSP	Zabezpečené jednoduché párování - Secure Simple Pairing
OOB	Mimo pásmo - Out-of-band
PII	osobní údaje - personally identifiable information
WPA	Chráněný přístup Wi-Fi - Wi-Fi Protected Access
WPS	Nastavení chráněné Wi-Fi - Wi-Fi Protected Setup
DFU	Aktualizace firmwaru zařízení - Device Firmware Update
OOK	On-Off Keying
ASK	Klíčování s amplitudovým posunem - Amplitude-Shift Keying
PAN	Osobní síť - Personal Area Network
GUI	Grafické uživatelské rozhraní - Graphic User Interface
ARM	Advanced RISC Machines

GOT	Tabulka globálního posunu - Global Offset Table
ECB	Elektronická kniha kódů - Electronic Code Book
CRC	Cyklická redundanční kontrola - Cyclic redundancy check
JSON	Objektový zápis jazyka JavaScript - JavaScript Object Notation
SSID	Identifikátor sady služeb - Service Set Identifier

A Seznam řetězců (Příloha)

```
----------------------
BARVY

cervena    330502ff000000ffc184000000000000000000071
zelena     33050200ff0000ffc184000000000000000000071
modra      3305020000ff00ffc184000000000000000000071

bila       330502ffffff00ffc184000000000000000000071

zluta      330502ffff0000ffc18400000000000000000008e
azurova    33050200ffff00ffc18400000000000000000008e
ruzova     330502ff00ff00ffc18400000000000000000008e

fialova         3305027f00ff00ffc184000000000000000000000e
svetle modra    330502007fff00ffc184000000000000000000000e
oranzova        330502ff7f0000ffc184000000000000000000000e
ruzova          330502ff007f00ffc184000000000000000000000e
azurova         33050200ff7f00ffc184000000000000000000000e
zluto-zelena    3305027fff0000ffc184000000000000000000000e

fialova         3305028b00ff00ffc184000000000000000000000fa
svetle modra    330502008bff00ffc184000000000000000000000fa
oranzova        330502ff8b0000ffc184000000000000000000000fa
ruzova          330502ff008b00ffc184000000000000000000000fa
azurova         33050200ff8b00ffc184000000000000000000000fa
zluto-zelena    3305028bff0000ffc184000000000000000000000fa

----------------------
JAS

  100%
          3304fe0000000000000000000000000000000000c9
          3304e7000000000000000000000000000000000d0
          3304ce0000000000000000000000000000000000f9
          3304c2000000000000000000000000000000000f5
          3304c1000000000000000000000000000000000f6
          3304b1000000000000000000000000000000000086
```

```
            330499000000000000000000000000000000ae
            330498000000000000000000000000000000af
            330495000000000000000000000000000000a2
            330494000000000000000000000000000000a3
            33048a000000000000000000000000000000bd
            330479000000000000000000000000000004e
            330466000000000000000000000000000000051
            330465000000000000000000000000000000052
            33045a000000000000000000000000000006d
            330450000000000000000000000000000000067
            33044c000000000000000000000000000007b
            330447000000000000000000000000000000070
            330445000000000000000000000000000000072
            330442000000000000000000000000000000075
            330434000000000000000000000000000000003
            330426000000000000000000000000000000011
            330423000000000000000000000000000000014
            330417000000000000000000000000000000020
            330416000000000000000000000000000000021
            330415000000000000000000000000000000022
    1%
----------------------
TEPLOTA bile

  tepla
    330502ffffff01ff8d0b00000000000000000b3  max
    330502ffffff01ff98360000000000000000009b
    330502ffffff01ffb16500000000000000000e1
    330502ffffff01ffd0970000000000000000072
    330502ffffff01ffdbba000000000000000054
    330502ffffff01ffead3000000000000000000c
    330502ffffff01fff2ec00000000000000002b
    330502ffffff01fef9ff00000000000000032
    330502ffffff01f3f2ff00000000000000034
    330502ffffff01e6ebff00000000000000038
    330502ffffff01dde6ff0000000000000000e
    330502ffffff01d6e1ff00000000000000002  max
  studena
```

```
--------------------
MOD

relax (color)   3305040e000000000000000000000000000003c
cteni (color)   3305040d000000000000000000000000000003f
zivy rezim      3305040c000000000000000000000000000003e

barevne (active) 3305040b000000000000000000000000000039

------------------------------------------------------------
ON    330101000000000000000000000000000033
OFF   330100000000000000000000000000000032
------------------------------------------------------------
```