VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky
a komunikačních technologií

BAKALÁŘSKÁ PRÁCE

Brno, 2020 Viktoriia Zhukova

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

HONEYPOT PRO PROTOKOL LORAWAN
HONEYPOT FOR LORAWAN PROTOCOL

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — **Viktoriia Zhukova**
AUTHOR

VEDOUCÍ PRÁCE — **Ing. Ondřej Pospíšil**
SUPERVISOR

BRNO 2020

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Bakalářská práce

bakalářský studijní program **Informační bezpečnost**

Ústav telekomunikací

Studentka: Viktoriia Zhukova ***ID:*** 203459

Ročník: 3 ***Akademický rok:*** 2019/20

NÁZEV TÉMATU:

Honeypot pro protokol LoRaWAN

POKYNY PRO VYPRACOVÁNÍ:

Student se v rámci práce zaměří na problematiku honeypotů v rámci IoT. Provede analýzu útoků na IoT zařízení a to jak obecných, tak i útoků pomocí např. botnetu. Následně se zaměří především na technologii LoRaWAN. Student následně vytvoří fyzický honeypot s možností volby míry interakce (nízká/vysoká). Tento honeypot bude využívat vlastní hardware (LoRaWAN gateway postavena např. na RaspBerry Pi a koncetrátoru ic880a). Honeypot bude monitorovat aktivitu útočníka a bude generovat patřičné logy. Vzhledem k tomu, že se jedná o fyzický HoneyPot s možností vysoké míry interakce, musí student tento HoneyPot dostatečně zabezpečit proti zneužití útočníkem a také proti jeho zničení (z tohoto důvodu budou zkoumány možnosti využití sandboxingu). Výsledkem tak bude bezpečný honeypot pro IoT protokol LoRaWAN s vysokou mírou interakce.

DOPORUČENÁ LITERATURA:

[1] „LoRaWAN What is it?: A technical overview of LoRa and LoRaWAN.“ LoRa Alliance. 2015.

[2] PROVOS, Niels a Thorsten HOLZ. Virtual honeypots: from botnet tracking to intrusion detection. Upper Saddle River, NJ: Addison-Wesley, c2008. ISBN 9780321336323.

Termín zadání: 3.2.2020 ***Termín odevzdání:*** 8.6.2020

Vedoucí práce: Ing. Ondřej Pospíšil

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

ABSTRAKT

Bakalářská práce se v teoretické části zabývá technologií LoRaWAN, popisuje komunikaci v síti. Věnuje se bezpečnosti LoRaWAN sítě a popisuje zranitelnosti a mitigační opatření. Druhá polovina teoretické části je zaměřena na popsání technologie typu honeypot, popisuje rozdělení honeypotů, uvádí jejich výhody a nevýhody a porovnává IoT honeypoty. Praktická část je pak zaměřena na sestavení experimentálního prostředí, je uveden popis zvoleného hardware a software. Popisuje se zprovoznění sítě LoRaWAN, konfigurace brány a spuštění serveru LoRaWAN. Dále je popsán celý postup sestaveni honeypotu. Jsou popsány a zkoumány dvě varianty high-interaction honeypotu. Jedna bez využití sandboxingu, s využitím firewall a druhá s využitím sandboxingu realizovaného pomocí chroot/jail.

KLÍČOVÁ SLOVA

honeypot, detekce útoků, sandboxing, IoT, LoRa®, LoRaWAN™, brána

ABSTRACT

The bachelor's thesis in the theoretical section focuses on LoRaWAN technology. It describes network communication, LoRaWAN security, vulnerabilities, and mitigation measures. The second half of the theoretical section focuses on describing honeypot technology and its distribution, listing its advantages and disadvantages, and comparing IoT honeypots. The practical section focuses on building an experimental environment. There is a description of the selected hardware and software. It describes the commissioning of the LoRaWAN network, the configuration of the gateway, and the startup of the LoRaWAN server. Next, the whole procedure of honeypot assembly is described. Two variants of high-interaction honeypot are described and investigated. One without the use of sandboxing and using a firewall, the other with the use of sandboxing and an implementation of chroot/jail.

KEYWORDS

honeypot, detection of attacks, sandboxing, IoT, LoRa®, LoRaWAN™, gateway

ZHUKOVA, Viktoriia. *Honeypot pro protokol LoRaWAN*. Brno, 2020, 53 s. Bakalářská práce. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací. Vedoucí práce: Ing. Ondřej Pospíšil

PROHLÁŠENÍ

Prohlašuji, že svou bakalářskou práci na téma „Honeypot pro protokol LoRaWAN“ jsem vypracovala samostatně pod vedením vedoucího bakalářské 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 autorka uvedené bakalářské práce dále prohlašuji, že v souvislosti s vytvořením této bakalářské práce jsem neporušila autorská práva třetích osob, zejména jsem nezasáhla nedovoleným způsobem do cizích autorských práv osobnostních a/nebo majetkových a jsem si plně vědoma 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 autorky

PODĚKOVÁNÍ

Rád bych poděkovala panu Ing. Radku Fujdiakovi, Ph.D za odborné vedení, konzultace, trpělivost a podnětné návrhy k práci. Dále bych chtěla poděkovat své rodině za podporu.

Obsah

Seznam obrázků

Seznam tabulek

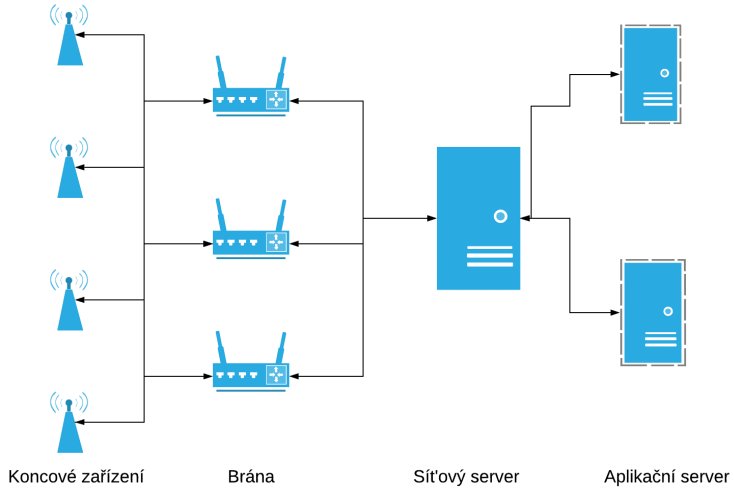
Úvod

Sítě LoRaWAN patří do kategorie sítí nízkého výkonu LPWAN (Low Power Wide Area Network) [1], které jsou založeny na metodě modulace LoRa, vyvinuté Semtech Corporation, také na otevřeném protokolu LoRaWAN, vyvinutém výzkumným centrem IBM Research ve spolupráci s Semtech Corporation. LoRa je vedoucí technologií LPWAN. Bezpečnost je primárním požadavkem pro jakýkoliv vývoj v oblasti internetu věcí. Honeypot je počítač nebo počítačový systém, který napodobuje pravděpodobné cíle kybernetických útoků [2]. Může být použit k detekci útoků nebo k jejich odklonu od legitimního cíle. Další možností použití je získání informací o tom, jak kyberkriminalisté fungují. Použití honeypot technologie pomáhá snížit rizika neznámých útoků a vnitřních úniků informací.
První část práce se zabývá technologií LoRaWAN, popisuje komunikaci v síti. Věnuje se bezpečnosti LoRaWAN a popisuje zranitelnosti a mitigační opatření.
Druhá část práce popisuje technologie typu honeypot, popisuje rozdělení honeypotů, uvádí jejich výhody a nevýhody a porovnává IoT honeypoty.
Ve třetí části je popsán návrh experimentálního prostředí, uveden popis zvoleného hardware a software. Popisuje se zprovoznění sítě LoRaWAN, konfigurace brány a spuštění serveru LoRaWAN. Následovně je popsáno vytváření honeypotu využívajícího vlastní hardware (LoRaWAN gateway postavená na RaspBerry Pi a koncetrátoru ic880a) s možností volby míry interakce (nízká/vysoká). Jsou popsány a zkoumány dvě varianty high-interaction honeypotu. Jedna bez využití sandboxingu s využitím firewallu a druhá s využitím sandboxingu realizovaného pomocí chroot/jail. Výsledkem práce je honeypot pro IoT protokol LoRaWAN s volbou míry interakce - vysoká/nízká.

1 Technologie LoRaWAN

1.1 Stručný technický popis

LoRaWAN definuje komunikační protokol a systémovou architekturu pro síť, zatímco fyzická vrstva Lora umožňuje dálkové komunikační spojení. Protokol a síťová architektura mají největší vliv při určování životnosti baterie uzlů, kapacity sítě, kvality služeb, zabezpečení a rozmanitost aplikací poskytovaných sítí [3].
Otevřený protokol LoRaWAN vyvíjí LoRa Alliance pro sítě s vysokou kapacitou (až 1 000 000 zařízení v jedné síti), s velkým dosahem (až 10-15 km v otevřeném prostoru) a s nízkou spotřebou energie. Protokol LoRaWAN poskytuje kompletní obousměrnou komunikaci mezi uzly sítě a má speciální metody šifrování pro zajištění spolehlivosti a bezpečnosti systému.
Standardní síť LoRaWAN může být prezentována jako koncová zařízení (body, uzly), data, která jsou přenášena (v zašifrované podobě) na brány, dále na server sítě poskytovatele a na aplikační server poskytovatele, odkud se vše přenáší ke koncovému uživateli obr. 1.1.



Obr. 1.1: Síť LoRaWAN

Koncová zařízení sítě LoRaWAN mohou plnit různé funkce, například měření, řízení a kontrolu. Obvykle jsou tyto uzly umístěny vzdáleně a všechny jsou napájeny bateriemi. Pomocí protokolu LoRaWAN jsou tato koncová zařízení konfigurována pro komunikaci s bránou LoRa. Data jsou přenášena obousměrně z uzlu na server a zpět. Uzly pracují v režimu přenosu pouze v krátkých intervalech, pak se otevře

dočasné okno pro příjem dat. Zbytek času jsou uzly buď v režimu spánku nebo ve stavu příjmu, který závisí na třídě zařízení:

- **Zařízení třídy A:** Koncové zařízení této třídy přenáší data do brány po částech podle daného plánu. Iniciátorem výměny je samotné koncové zařízení. Uzel obvykle nevyžaduje potvrzení o obdržení své zprávy, nicméně některé aplikační servery tvoří zvláštní odpověď, "potvrzení", a síťový server vybere nejlepší trasu (gateway) pro odeslání potvrzení v okamžiku otevření uzlu přijímacího okna (zpráva s potvrzením). Uzel (end-node) přejde do režimu příjmu (otevře se okno přijímání) na krátkou dobu po odeslání dat. V klidu, po zbytek času, je v režimu úspory energie nebo spánku (sleep). Server hromadí pro uzly (end- node) zprávy a předá je okamžitě, když se připojí koncové zařízení. Tato třída koncových zařízení se často využívá v praxi.
- **Zařízení třídy B:** Koncové zařízení této třídy má možnost plánování přijímacích period, které jsou určeny serverem. Server odesílá zprávy uzlu podle plánu. Iniciátorem výměny může být server LoRaWAN sítě. Zařízení této třídy synchronizují vnitřní čas s časem sítě pomocí řídících rámců (z angličtiny. beacon), které pravidelně přijímají od brány. Uzly v této třídě mají relativně nízké časové zpoždění při sdílení dat a otevírají širší časové okno příjmu, ve srovnání s třídou A. Koncová zařízení třídy B mají také všechny funkce zařízení třídy A.
- **Zařízení třídy C:** Koncová zařízení této třídy mají přijímací okno otevřeno nepřetržitě a zavírají pouze po dobu krátkodobého přenosu dat. Server může zahájit výměnu kdykoliv a předat zprávy uzlu okamžitě, jak se objeví. Tato třída zařízení spotřebuje největší množství energie (ve srovnání s třídami A a B) a přijímá data od serveru LoRaWAN sítě s nejmenším zpožděním. Zařízení třídy C mají všechny funkce zařízení třídy A a B [4].

1.2 Bezpečnost LoRaWAN

Jedním z nejzranitelnějších míst v síti LoRaWAN je fáze připojení nových zařízení do sítě. Pro zajištění bezpečnosti sítí obecně, je nutné zajistit bezpečnost v této fázi aktivace. Aby koncové zařízení bylo připojeno do sítí je potřeba provést jeho aktivaci. Tu lze provést dvěma způsoby.

1.2.1 Aktivace vzduchem

Při použití této metody na straně koncového zařízení a na straně serveru musí být zadány tři parametry: jedinečný identifikátor koncového zařízení (DevEUI), identifikátor serveru (AppEUI) a klíč serveru (AppKey). To znamená, že server by měl

zpočátku znát zařízení, které se k němu pokusí připojit. Na začátek koncové zařízení vysílá paket join_request na jednu ze tří předem stanovených frekvencí. Tímto paketem se ptá, jestli je v blízkosti síť, která ho "pozná". Níže tab. 1.1 je složení balíčku join_request, který obsahuje DevEUI, AppEUI a DevNonce.

Tab. 1.1: Složení balíčku join_request.

Velikost(bytes)	8	8	8
Join Request	AppEUI	DevEUI	DevNonce

DevNonce je náhodná veličina. Server ji nějaký čas udržuje v paměti a když přijde join_request se stejným DevNonce, jako jeden z předchozích, server tento požadavek zamítne. To se používá pro ochranu proti “replay” útoku, kdy útočník může zaznamenat požadavek na aktivaci a pak jej opakovat ze svého zařízení.
Join_request se přenáší nešifrovaným způsobem. Join_accept přijde v případě, že na serveru jsou známé AppEUI a DevEUI a stejně tak není zaznamenán DevNonce. Jinak odpověď nebude následovat. Pokud je vše v pořádku zkontrolované, server generuje odpověď join_accept. Jeho složení je uvedeno níže tab. 1.2.

Tab. 1.2: Složení balíčku join_accept

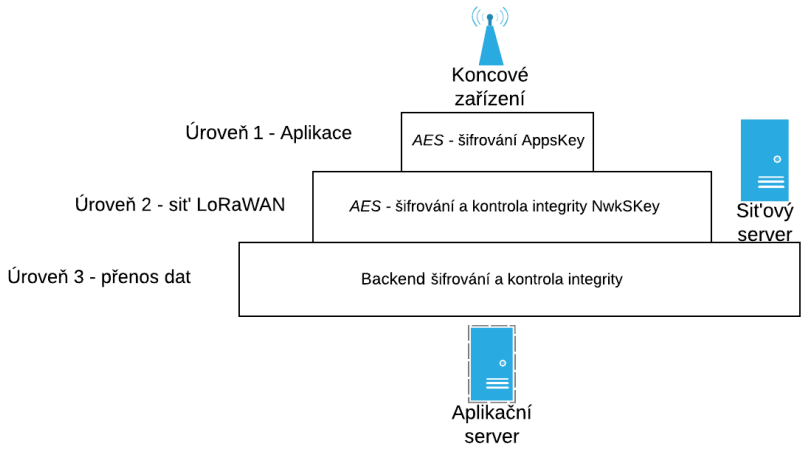
Velikost(bytes)	3	3	4	1	1	16(Optional)
Join Accept	AppNonce	NetID	DevAddr	DLSettings	RxDelay	CFList

Jsou generovány dva klíče síťového serveru (NwkSKey) a aplikačního serveru (AppSKey). Tyto klíče spolu s dalšími informacemi jsou šifrovány klíčem AppKey a odesílány koncovým zařízením. Dále výměna zpráv probíhá s dvojím šifrováním relačních klíčů. NwkSKey se nepoužívá pro šifrování paketů pouze s daty (bez příkazů), ale je používán pro kontrolní součet. NwkSkey a AppSKey jsou jedinečné pro každé koncové zařízení [5].

1.2.2 Aktivace podle přednastavení

Při použití této metody jsou DevAddr, NwkSKey a AppSKey uloženy přímo do zařízení. Výhoda této metody spočívá v tom, že není potřeba provádět aktivaci, koncové zařízení je okamžitě připraveno k práci. Nevýhodou této metody je to, že

není bezpečná [5]. Síť LoRaWAN využívá víceúrovňový systém zabezpečení přenosu dat obr. 1.2.



Obr. 1.2: Systém zabezpečení přenosu dat

- **Úroveň 1.** - AES Šifrování na úrovni aplikace (mezi koncovým zařízením a aplikačním serverem) pomocí 128 bitového relačního klíče AppSKey. Tento klíč je uložen na koncovém zařízení a na aplikačním serveru a není k dispozici provozovateli sítě. Přístup k AppSKey má pouze vlastník aplikačního serveru.
- **Úroveň 2.** - AES šifrování a kontrola integrity zpráv na síťové vrstvě (mezi koncovým zařízením a síťovým serverem) pomocí 128-bit klíče NwkSKey. Tento klíč je uložen na koncovém zařízení a na síťovém serveru a není k dispozici klientovi. Přístup k NwkSKey je k dispozici pouze vlastníkům síťového serveru.
- **Úroveň 3.** - Standardní metody ověřování a šifrování internet protokolu (IPsec, TLS) při přenosu dat přes síť mezi uzly sítě (koncové zařízení, síťový server, Join_server, aplikační server).

Aplikačním nebo síťovým serverem v každém okamžiku může být uskutečněn přechod na novou relaci s generováním nové sady klíčů šifrování, takže staré šifrovací klíče už nebudou platné. Také existuje možnost stanovení režimu periodického generování nové sady klíčů NwkSKey a AppSKey. Ve verzi standardu LoRaWAN V1.0.x se klíč AppSKey ukládá na straně síťového serveru. Na tomto typu serveru probíhá i vytvoření relace klíčů, nicméně ve verzi V1.1 pro tyto účely vzniká zvláštní server (Join_server). Join_server může být dále chráněn samostatným bezpečnostním modulem HSM (Hardware Security Module).

V případě Join_serveru jsou zavedeny další klíče pro bezpečný přenos generovaných klíčů mezi servery, stejně jako jejich ukládání na síťový server a aplikační server: AS_Key pro klíč AppSKey a LRC_K pro klíč NwkSKey. Na koncovém zařízení

se šifrovací klíče mohou bránit speciálním hardwarovým prvkem bezpečnosti (např. Microchip ATECC608A), což znemožní jejich napadení v případě fyzického vlivu na terminál. Implementace hardwarových prostředků do sítě a na koncové zařízení znemožní pokusy o zachycení relace klíčů při jejich přenosu mezi servery a pokus o útok na síťový server [6].

1.3 Aktiva v LoRaWan síti

Cíl útoku je definován ve 2 aspektech: ohrožení bezpečnostních vlastností sítě a ohrožení aktiva síťového zabezpečení. Bezpečnostní vlastnosti v síti se vždy popisují pomoci trojice CIA (Confidentiality, Integrity, and Availability). CIA trojice, která zahrnuje důvěrnost, integritu a dostupnost, odráží důležité požadavky na bezpečnost informací. Za účelem členění útoku na síť LoRaWAN je důležité zvážit tyto tři prvky. Bezpečnostní prostředky odrážejí důležité parametry v síti. Pokud jsou tyto parametry ohroženy, bezpečnost sítě bude taktéž ohrožena. V tab. 1.3 se uvádí prostředky, které jsou chráněny v sítích LoRaWAN a klasifikace prostředků podle důležitosti.
Například, NwkSKey je primární aktivum, protože útok, který by mohl ohrozit NwkSKey je schopen přímo ohrozit síť pomocí klíče pro dešifrování zprávy. Pro sekundární aktivum, jako je AppKey, útočník získá pouze položku DevNonce, ale potřebuje více prostředků pro útok na síť.
Některá aktiva v tabulce jsou důvěrná, zatímco jiná nejsou. Například důvěrnost AppNonce je chráněna, zatímco DevNonce není. Je to proto, že k odvození dvou relačních klíčů je potřeba použití obou těchto položek. Pokud nejsou chráněny, relační klíče lze odvodit, jakmile útočník získá AppKey.
Popis aktiv:

- **NwkSKey:** NwkSKey se používá na síťovém serveru ke kontrole podpisu zprávy. Je generován během aktivace uzlu. Pokud je důvěrnost NwkSKey ohrožena, třetí strana může použít NwkSkey k vygenerování své vlastní zprávy v LoRaWAN síti, a projít procedurou kontroly podpisu na síťovém serveru. Pokud je integrita NwkSkey ohrožena na síťovém serveru nebo na koncovém zařízení, komunikační relace zařízení bude ohrožena, a všechny zprávy v této komunikační relaci mohou být zahozeny, protože neprojdou proceduru ověření.
- **AppSkey:** AppSKey se používá na aplikačním serveru k dešifrování zpráv. Generuje se během aktivace uzlu. Pokud je důvěrnost AppSKey ohrožena, útočník bude schopen dešifrovat všechny zprávy. Důvěrnost celé sítě LoRaWAN bude ohrožena. Při ohrožení integrity AppSKey aplikační server nebo koncové zařízení nebudou schopny dešifrovat zprávy správně. Z toho vyplývá, že získaná data nemohou být důvěryhodná.

- **AppKey:** AppKey se používá při aktivaci uzlů pro zařízení, která jsou aktivována pomocí metody aktivace vzduchem (Over-the-Air-Activation) a slouží pro odvození dvojice klíčů AppSKey a NwkSKey. Je přiřazen vlastníkem aplikace před aktivací na zařízení a na server. Pokud je důvěrnost AppKey ohrožena, útočník bude schopen realizovat "replay" útok. Z toho vyplývá, že bude schopen připojit škodlivé zařízení k síti. Pokud je ohrožena integrita aplikace AppKey, koncové zařízení se nebude moci připojit k síti LoRaWAN prostřednictvím aktivace vzduchem (Over-the-Air-Activation).
- **DevNonce:** DevNonce je generována koncovým zařízením. Při aktivaci vzduchem (Over-the-Air-Activation) bude tato položka přenášena z koncového zařízení do brány a síťového serveru a poté budou položky DevNonce a AppNonce šifrovány pomocí položky AppKey, která může vypočítat NwkSKey a AppSKey a pak ohrozit bezpečnost celé LoRaWAN sítě. DevNonce lze přenášet v otevřeném textu, protože bez klíče AppSkey nemůže útočník dosáhnout žádných útoků. Integrita DevNonce by měla být chráněna. Bez ochrany integrity jsou klíče relace, generované koncovým zařízením a serverem, odlišné. Komunikační relace bude neplatná.
- **AppNonce:** AppNonce je určitá forma jedinečného ID (identifikátoru) síťového serveru. Používá se pro generování klíčů relace pomocí položek DevNonce a AppKey. V případě, že důvěrnost AppNonce není chráněna a útočník získá AppKey, může snadno vypočítat NwkSKey a AppSkey a narušit bezpečnost celé LoRaWAN sítě. Bez ochrany integrity jsou klíče relace, generované koncovým zařízením a serverem, odlišné. Komunikační relace bude neplatná.
- **FrmPayload:** FrmPayload se skládá z přenášených dat. Například, data teplotního senzoru jsou přenášena pomocí položky FrmPayload. Pokud útočník ohrozí důvěrnost FrmPayload, data ze senzorů budou známá útočníkům a to způsobí problémy s ochranou soukromí.
- **DevAddr:** DevAddr je identifikátor koncového zařízení. Je v otevřeném textu. Pokud je jeho integrita ohrožena, bude komunikace mezi koncovým zařízením a serverem přerušena. Data senzorů přijímaná serverem nebudou platná a neměla by být důvěryhodná.
- **FCnt:** FCnt je hodnota čítače v koncovém zařízení a na serveru. Je v otevřeném textu. Pokud bude narušena integrita FCnt, bude obtížné udržovat synchronizaci pro server a koncové zařízení. Kvůli změně hodnoty čítače útočníci snadno dosáhnou "replay" útoku.
- **ACK:** ACK je speciální parametr zprávy, který se používá k potvrzení přijetí zprávy. Je v otevřeném textu. Pokud je narušena integrita ACK, je možné, že zpráva ACK bude upravena na normální zprávu a funkce ACK bude zakázána.
- **MAC příkazy:** Příkazy MAC lze odesílat ve složkách FOpts nebo FrmPay-

load. Některé příkazy, například nastavení parametrů rádia, by měly být důvěrné. Jinak útočník může snadno zjistit stav rádia koncového zařízení. Integrita MAC příkazů by také měla být chráněna [7].

Tab. 1.3: Aktiva v LoRaWAN síti.

Aktivum	Primární aktivum	Sekundární aktivum	Integrita	Důvěrnost
NwkSKey	ano	ne	ano	ano
AppSKey	ano	ne	ano	ano
AppKey	ne	ano	ano	ano
DevNonce	ne	ano	ano	ne
AppNonce	ne	ano	ano	ano
FrmPayload	ano	ne	ano	ano
DevAddr	ne	ano	ano	ne
Fcnt	ne	ano	ano	ne
ACK	ne	ano	ano	ne
MAC příkazy	ne	ano	ano	ano

1.4 Popis zranitelnosti sítí LoRaWAN a mitigační opatření

V sítích LoRaWAN je útočník schopen:

- prozkoumat síť a zařízení LoRa
- zachycovat a posílat zprávy vzduchem
- zpracovávat a ukládat data
- pokud zná klíče, je schopen šifrovat a dešifrovat
- mít fyzický přístup

Protože cílem práce je vytváření honeypotu na bráně, následovně v tab. 1.4 budou popsány zranitelnosti zaměřené na bránu a jejich detekce. Pro sestavení tabulky byly použity materiály z [9].

Níže je uveden popis různých útoků nebo kybernetických hrozeb, které jsou aktuální pro IoT síť:

- **Spoofing(ARP/ DNS/IP/ACK):** pomocí škodlivého programu/zařízení útočník vydává sebe za prvek v Iot síti (server, brána, koncové zařízení) a padělá (falsifikuje) přenášená data. Díky tomu realizuje útok na síť, šíří malware a přistupuje k důvěrným informacím a datům.

Tab. 1.4: Zranitelnosti zaměřené na bránu

Název	Detekce	Vrstva
Neoprávněný přístup (Unauthorized access)	Monitorování IoT zviditelňuje přenášené příkazy IoT, což pomáhá detekovat neautorizované příkazy.	fyzická
Selfish threat (selhání koncových zařízení/bran/serveru)	Monitoring sítí	aplikační, prezentační, relační, transportní, síťová, linková, fyzická
Man in the middle (muž uprostřed)	Detekce komunikace mimo standardní doby (těžké detekovat)	aplikační, prezentační, transportní, síťová
Spoofing attack (odposlech)	Omezení výkonu, pravděpodobnost výpadku utajení a pravděpodobnost úniku (obtížné zjistit)	fyzická, linková, síťová
Prohledávání bannerů (network mapping)	Analýza provozu, detekce anomálií	linková, síťová
Škodlivý kód	Analýza provozu, detekce anomálií	aplikační
Útok hrubou silou	Detekování více neúspěšných pokusů o přihlášení pro jednoho uživatele v krátkém časovém úseku	transportní
Dos/DDos (amplifikační útok)	Zvýšení provozu, neobvyklý pravidelný provoz, podpis pro konkrétní útoky	aplikační, prezentační, relační, transportní, síťová, linková, fyzická
Hrozby přenosu dat (Data transmission threats)	Monitoring sítí	aplikační, prezentační, relační, transportní, síťová, linková, fyzická

- **Botnety:** IoT botnet je ten, ve kterém jsou kompromitována nekonvenční internetová zařízení. Patří mezi ně routery, kamery a podobná zabudovaná zařízení, která jsou hacknuta virem. Virus pak usnadňuje kybernetickým zločincům převzít kontrolu nad těmito zařízeními, podobně jako tradiční botnet. Po přidání zařízení IoT do botnetu se neustále snaží vyhledávat nové oběti, dokud nejsou znovu zajištěny. Obvykle může tradiční botnet obsahovat zařízení v tisících, zatímco IoT past je mnohem větší - stovky tisíc zařízení mohou být ohroženy jedinou botnetovou kampaní. Zařízení připojená k internetu, infikovaná a ohrožená malwarem, která společně pracují pod společnou kontrolou, se většinou používají unisono pro generování masivních útoků DDoS od více klientů [8].
- **Distribuované zamítnutí služby (DDoS):** Útok, který se pokouší narušit nebo učinit službu online nedostupnou, a to tak, že ji převede z více (distribuovaných) zdrojů.
- **C2:** Příkazový a řídicí server, který zařazuje příkazy do botnetů.
- **Prohledávání bannerů:** Technika obvykle používaná k provádění inventury systémů v síti, kterou může útočník také použít k získání informací o potenciálním cíli útoků provedením požadavků HTTP a kontrolou vrácených informací operačního systému (OS) a počítače (například nc www.target.com 80).
- **Brute force:** Metoda pokusu a omylu pro získání přístupu k systému nebo obcházení šifrování.
- **Přetečení vyrovnávací paměti:** Využívá chybu nebo defekt ve spuštěném softwaru, který jednoduše přeteče vyrovnávací paměť nebo blok paměti s více daty, než je přiděleno. Toto překročení může zapisovat přes jiná data na sousedních adresách paměti. Většina chyb přetečení je výsledkem špatně zkonstruovaného softwaru, který nekontroluje hranice vstupních hodnot. Zařízení internetu věcí obvykle potřebují šetřit energií, což vede k malému množství energeticky účinné paměti. Čím menší je vyrovnávací paměť, tím snazší je přetečení, díky čemuž je IoT perfektním prostředím pro tyto druhy útoků.
- **Útok na korelaci výkonu:** Umožňuje odhalit tajné šifrovací klíče uložené v zařízení pomocí čtyř kroků a to jsou prozkoumat dynamickou spotřebu energie cíle a zaznamenat jej pro každou fázi normálního šifrovacího procesu. Dále vynutit cíl zašifrovat několik objektů prostého textu a zaznamenat jejich spotřebu energie. Poté zaútočit na malé části klíče (podklíče) zvážením každé možné kombinace a vypočtením Pearsonova korelačního koeficientu mezi modelovanou a skutečnou silou. Nakonec sestavit nejlepší podklíč k získání úplného klíče.
- **Slovníkový útok:** Metoda získávání vstupů do síťového systému systematickým zadáváním slov ze souboru slovníku obsahujícího páry uživatelských jmen

a hesel.

- **Fuzzing:** Útok, který spočívá v odeslání vadných nebo nestandardních dat do zařízení a pozorování, jak zařízení reaguje. Například pokud zařízení pracuje špatně nebo vykazuje nepříznivé účinky, tento fuzz útok by mohl odhalit slabost.
- **Útok typu Man-in-the-middle (MITM):** Běžná forma útoku, která umístí zařízení do středu komunikačního proudu mezi dvěma netušícími stranami. Zařízení poslouchá, filtruje a přizpůsobuje informace z vysílače a znovu vysílá vybrané informace do přijímače. MITM může být ve smyčce jako opakovač nebo může být postranním pásmem, který poslouchá přenos, aniž by zachytil data.
- **Replay útok:** Síťový útok, známý také jako útok při přehrávání, kdy data jsou zlomyslně opakována nebo přehrávána původcem nebo protivníkem, který zachycuje data, ukládá je a podle přání je přenáší.
- **Využití RCE:** Vzdálené spuštění kódu, které útočníkovi umožňuje provádět libovolný kód. Obvykle se jedná o útok přetečení vyrovnávací paměti přes HTTP nebo jiné síťové protokoly, které injektují malware kód.
- **Return-to-libc:** Typ útoku, který začíná přetečením vyrovnávací paměti, kde útočník vloží kód, aby skočil do libc nebo do jiných populárně používaných knihoven v paměťovém prostoru procesů při pokusu o přímé vyvolání systémových rutin. Obchází ochranu poskytovanou nevykonatelnými paměťovými a ochrannými pásmy. Toto je specifická forma útoku ROP.
- **Ransomware:** Ransomware je druh škodlivého softwaru, který po infikování počítače přebírá kompletní kontrolu nad systémem a uzavírá data. Po odepření přístupu požaduje útočník výkupné výměnou za přístup k údajům. Uživatelé pak mají průvodce, jak zaplatit výkupné a jak mohou použít dešifrovací klíč k odemknutí uzamčených dat.
- **Rootkit:** Obvykle škodlivý software (i když se často používá k odemykání smartphonů) používaný k tomu, aby umožnil nedetekovatelnost jiných softwarových dat. Rootkity používají několik cílených technik, jako jsou přetečení vyrovnávacích pamětí k útoku na služby jádra, hypervizory a programy uživatelského režimu.
- **Útok postranním kanálem:** Útok používaný k získávání informací ze systému oběti pozorováním sekundárních účinků fyzického systému, spíše než hledáním exploitů za běhu nebo exploitů za den. Příklady útoků postranních kanálů zahrnují analýzu korelační energie, akustickou analýzu a čtení zbytku dat po vymazání z paměti.
- **Sociální inženýrství:** útok založeny na psychologické manipulaci a osobním klamání k získání soukromých informací.

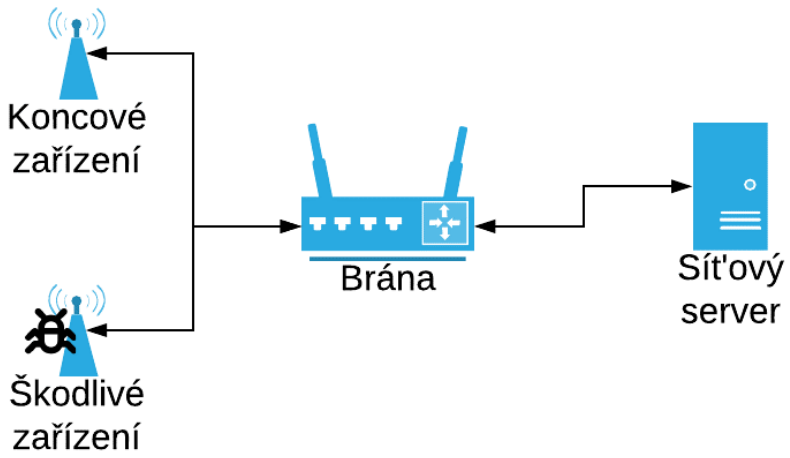
- **Záplava SYN:** Vyskytuje se, když hostitel odešle paket TCP SYN, který bude podvodný agent falšovat. To způsobí, že hostitel vytvoří polootevřená připojení k mnoha neexistujícím adresám, což hostovi vyčerpá všechny zdroje.
- **Nulové denní exploity:** Bezpečnostní vady nebo chyby v komerčním nebo produkčním softwaru, které nejsou známé vývojáři nebo výrobci [8].

1.4.1 Replay útok na zařízení aktivovaná pomocí ABP

ABP metoda aktivace používá klíče NwkSKey a AppSKey které jsou již uloženy do zařízení. Z toho vyplývá, že koncové zařízení, aktivované pomoci metody ABP znovu použije hodnotu čítače rámců od 0 pomocí stejného klíče. Fáze Replay útoku jsou následující:

1. Zachycení zpráv. Pomocí zařízení je potřeba zaznamenávat zprávy na uzlu, aktivovaného ABP metodou a uložit tyto zprávy do databáze.
2. Získaní hodnoty FCnt. Zjištění hodnoty čítače při vzestupném spojení.
3. Vyčkání, až se koncové zařízení resetuje nebo přeteče čítač.
4. Nalezení vhodné zprávy. Výběr zachycené zprávy s vhodnou hodnotou čítače z databáze útočníka.
5. Opakované odesílání zprávy na bránu.

Tento útok (obr. 1.3) může být velmi škodlivý pro koncová zařízení aktivovaná ABP metodou, ve velké LoRaWAN síti.



Obr. 1.3: Replay útok

V malé síti LoRaWAN s pouze několika koncovými zařízeními útočník pravděpodobně bude dlouho čekat na přetečení čítače. Ve velké síti LoRaWAN s více koncovými zařízeními je čekací doba na přetečení některého z koncových zařízení

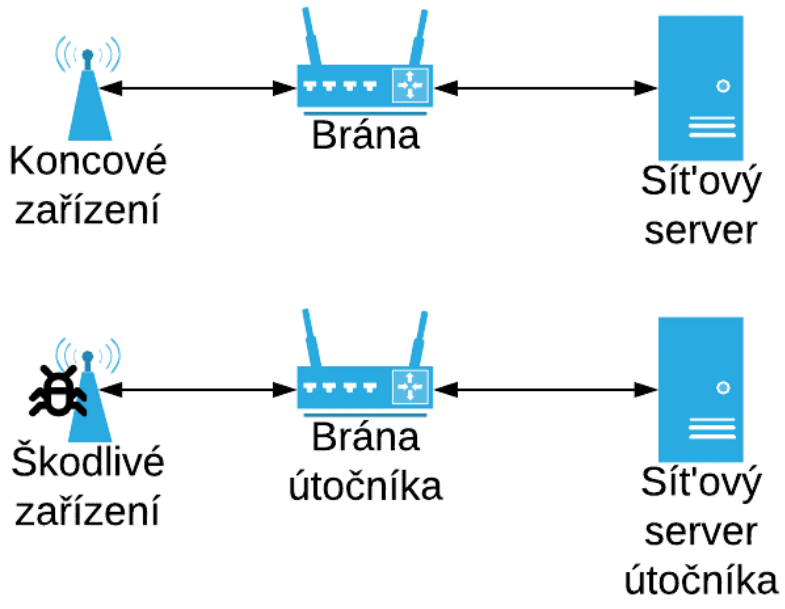
výrazně snížena. Jakmile útočník získá největší možnou hodnotu čítače pro jedno koncové zařízení, může tuto zprávu pravidelně opakovat a tím dosáhnout toho, že koncové zařízení bude trvale odepřeno. Pokud se nezmění klíče relací koncového zařízení, koncové zařízení nemůže znovu fungovat. Pokud útočník najde způsob, jak resetovat koncová zařízení (např. výpadek napájení), pak není potřeba, aby útočník čekal na přetečení čítače. Resetováním koncového zařízení a provedením replay útoku budou zprávy od postiženého koncového zařízení ignorovány.

Mitigační opatření:

1. Minimalizace použití ABP metody aktivace a nahrazení metodou OTAA.
2. Pokud bude použita metoda ABP:
   - Pravidelně měnit klíče NwkSKey a AppSKey.
   - Fyzické chránění koncových zařízení - tj. použití NVRAM *(Non-volatile random-access memory)* k ochraně čítače hodnot a nedovolení náhlé změny.
   - Kontrola maximální hodnoty čítače (Pokud je použita metoda OTAA: opakovat znovu aktivační proceduru) [7].

1.4.2 Odposlech

Důvod tohoto útoku (obr. 1.4) - bloková šifra (AES) vytváří přesně stejný klíčový materiál pokaždé, když se hodnoty čítače opakují.



Obr. 1.4: Odposlech

Útok může být proveden v následujících krocích:

1. Útočník zachycuje a ukládá bezdrátové pakety LoRaWAN a zaznamenává základní informace.
2. Po resetování pokračuje ve sběru paketů. Porovnává pakety před a po resetu. Spáruje pakety se stejnou hodnotou čítače.
3. Kódování metodou crib dragging.

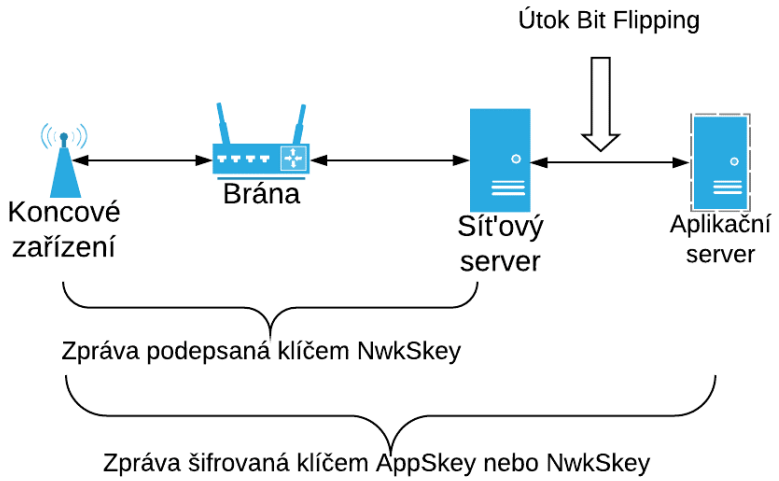
Útok je zaměřen na narušení šifrovací metody, kterou používá síť LoRaWAN. Zkoumáním bezdrátového provozu mezi branou a koncovým zařízením útočník může použít odpovídající vztah mezi dvěma zprávami se stejnou hodnotou čítače k dešifrování šifrovacího textu. Po útoku narušitel může ohrozit důvěrnost systému a získat data koncového zařízení přenášená v systému. V případě, že LoRaWAN přenáší tajná data, může tento útok způsobit vážné problémy se soukromím.

Mitigační opatření:

1. Nahrazení hodnoty čítače pro určitý čas hodnotou, která je generována z kryptograficky bezpečného generátoru pseudonáhodných čísel.
2. Změna klíče při resetování [7].

1.4.3 Útok Bit flipping

Cílem tohoto útoku (obr. 1.5) je prokázat, že integrita mezi síťovým serverem a aplikačním serverem není chráněna.



Obr. 1.5: Útok Bit flipping

Odeslané zprávy jsou šifrovány a poté podepsány. Jakmile jsou tyto zprávy obdrženy síťovým serverem, server použije NwkSKey pro kontrolu podpisu zprávy.

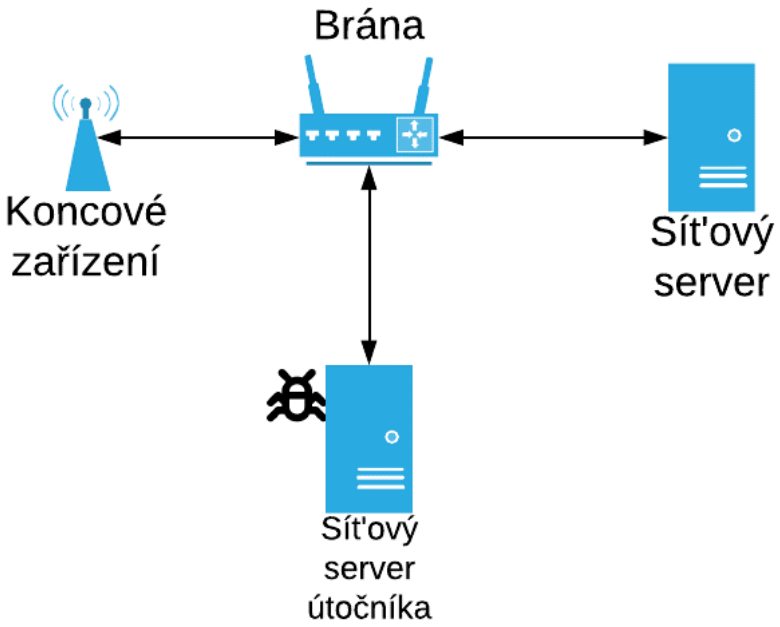
Následovně šifrované zprávy jsou potvrzeny na síťovém serveru a jsou odeslány na aplikační server. Data mezi síťovým a aplikačním serverem mohou být během této doby upravována, protože když se zprávy dostanou na aplikační server, integrita šifrované zprávy již nebude kontrolována.

Mitigační opatření:

1. Kontrola integrity na aplikačním serveru namísto síťového serveru
2. Změna pole protokolu: vyměnit CRC za MIC [7].

1.4.4 Útok ACK SPOOFING

Tento útok (obr. 1.6) je navržen tak, aby prokázal chybný návrh zprávy ACK v LoRaWAN síti.



Obr. 1.6: Útok ACK SPOOFING

Koncové zařízení očekává po odeslání zprávy potvrzovací zprávu ACK od síťového serveru. Pokud je zpráva ACK zachycena, tak bude zaslána útočníkem a nebude to správná zpráva ACK pro potvrzení obdržení zprávy. K dosažení tohoto musí být útočník schopen:

- mít kontrolu nad branou.
- rozpoznat zprávy ACK a zastavit přenos ACK zpráv sestupného spojení z brány na koncové zařízení podle potřeby.
- čtení ACK zpráv a výběru ACK s vhodnou hodnotou DevAddr a FCnt.

- odesílání vybraných zpráv ACK z brány na koncové zařízení.

Mitigační opatření: K útoku ACK SPOOFING dochází, protože ACK neurčuje, kterou zprávu ve skutečnosti potvrzuje. Aby nedocházelo k tomuto typu útoku, lze použít MIC (Message Integrity Code) na připojení k síťovému a aplikačnímu serveru. Následně je možné k vrácenému ACK přidat kryptografický kontrolní součet a následovné zařízení IoT může potvrdit, že ACK patří do této zprávy a ta zůstane během přenosu nezměněna [7].

1.4.5 Útok LoRa class B

V síti LoRaWAN třídy B se koncová zařízení pravidelně probouzejí a čekají na příchozí zprávy, jejichž trvání je určeno rámci vysílanými branou. Rámce nejsou šifrovány ani chráněny proti škodlivé úpravě. Protože neexistuje šifrování, všechny informace, které rámec obsahuje, jsou v otevřeném textu. Pokud jsou přenášena klíčová data, útočník je dokáže přečíst. CRC se používá k ochraně integrity důležité části rámce (Čas a NetID), ale závisí na parametrech fyzické vrstvy a útočník ji také může vypočítat. Pokud mají útočníci základní znalosti o BCNPayload, mohou vytvářet a odesílat vlastní škodlivý rámec se škodlivými parametry, a ten bude přijímán a zpracováván koncovými zařízeními.

Mitigační opatření: Změnit PHY a CRC na MIC, což povede k ověřování rámců [7].

1.4.6 Kompromitované zařízení a síťové klíče

Útočník s fyzickým přístupem může ohrozit koncová zařízení LoRa. Pokud útočník získá fyzický přístup k zařízení, může extrahovat klíče. Koncová zařízení obvykle obsahují rádiový modul LoRa a jednotku mikrokontroléru hostitele (MCU). Rádiový modul komunikuje s hostitelským mikrokontrolérem přes rozhraní UART nebo SPI. Škodlivý subjekt může zachytit všechny výměny dat mezi hostitelským MCU a rádiovým modulem a použít tyto intercepční informace k vytvoření simulovaného zařízení se stejnými přihlašovacími údaji nebo manipulujícího s datovým užitečným zatížením [10].

1.4.7 Rádiové rušení LoRaWAN

Rádiové rušení je jedním z vážných problémů pro internet věcí. Škodlivé entity mohou vysílat silný rádiový signál v blízkosti aplikačních zařízení a narušit rádiové přenosy. Tyto útoky obvykle vyžadují vyhrazený hardware, který minimalizuje možnost rušení útoků v reálných zařízeních. Praktická realizace útoku je popsána v [10].

1.4.8 Wormhole útok

Tento útok se provádí na fyzické vrstvě. Koncová zařízení v síti LoRaWAN mohou být zaseknuta pomocí běžného hardwaru. Dohromady s replay útokem lze proti provést wormhole útok v síti LoRaWAN. Při tomto typu útoku škodlivé zařízení zachytí pakety z koncového zařízení a vysílá na jiné vzdálené zařízení pro přehrání zachyceného paketu. To může snadno spustit nebezpečná entita bez předchozí znalosti sítě nebo kryptografie mechanismus [10].

1.4.9 Útok frame-delay

Koncové zařízení a brána nejsou poškozeny protivníkem. Protivník však může zpozdit dodávky uplinkových rámečků. Škodlivé zpoždění pro všechny uplink rámy je konečný. Kromě toho rámem nelze manipulovat díky kryptografické ochraně. Výsledkem útoku jsou nesprávná časová razítka pod synchronizací přístupu. Útok je podrobně popsán v [11].

2 Honeypot

Honeypot je hardwarový nebo virtualizovaný síťový prvek, který má být sondován, napaden nebo kompromitován útočníkem [12]. Honeypot se liší od klasických bezpečnostních nástrojů, jako jsou firewally či IDS (systém pro odhalení průniků) tím, že nejsou aplikovány pro konkrétní funkci. Honeypot je flexibilní prostředek, který lze aplikovat v různých situacích. Pomocí těchto prostředků lze umožnit prevenci nebo detekci útoků. V podstatě honeypot obsahuje některé funkce prakticky všech bezpečnostních nástrojů.
Mezi výhody tohoto prvku patří:

- shromažďování smysluplných informací.
- data, která byla skenována, popřípadě pozměněna, jsou následně zaznamenána a mají vysokou hodnotu.
- nenáročnost na systémové zdroje.
- honeypot zachycuje akce, které jsou zaměřeny pouze na něj, takže systém není přeplněný provozem.
- snadná instalace, konfigurace a provoz.

Za nevýhody lze považovat:

- omezená oblast vidění.
- největší nevýhodou prostředků honeypot je úzká oblast vidění. Honeypoty monitorují jen činnost, která je proti nim zaměřená. Pokud jsou akce útočníka zaměřeny na různé podsystémy sítě, nebude tuto činnost detekovat, pokud není zaměřena přímo na něj.
- riziko, že honeypot bude odhalen útočníkem – honeypot má určité očekávané vlastnosti nebo funkce [12] [13] [14].

2.1 Dělení prvků Honeypot

Existují různé způsoby rozdělení honeypotů (tab. 2.1). V této kapitole bude popsáno rozdělení honeypotů dle míry jejich interakce a také budou probrána různá kritéria, podle kterých se tyto prostředky rozdělují.

- dle úrovně protokolování
  Charakterizuje míru podrobností, s nimiž bude protokolování provedeno. Čím vyšší je úroveň protokolování, tím větší detaily mají záznamy protokolu programu. Nízká úroveň protokolování je určena malou úrovní podrobností shromážděných dat. Průměrná úroveň protokolování může zahrnovat protokol obě strany interakce, jakož i další údaje (např. konkrétní čas příchodu dat, ID interakcí). Vysokou úroveň protokolování mají honeypoty silné interakce, kdy

nástroj bere na sebe povinnost protokolovat všechny události, které probíhají v systému v interakci.

- dle úrovně simulace
  Charakterizuje stupeň simulace služby. Nejjednodušší úrovní simulování znamená prakticky úplnou absenci podpory funkce simulované služby (například možnost zobrazit pouze uvítací zprávu při připojení). Průměrná úroveň znamená poměrně podrobnou simulaci služby, s přihlédnutím k charakteristikám její práce. Vysoká úroveň simulace zahrnuje plnou implementaci všech funkcí služby (ve skutečnosti se blíží k emulaci služby).
- dle úrovně rizika
  Charakterizuje stupeň rizika při použití honeypotu. Čím více funkcí poskytuje honeypot, tím vyšší je pravděpodobnost, že honeypot může být použit pro útok jiných systémů nebo služeb.
- dle procesu instalace a konfigurace
  Toto kritérium charakterizuje časové a pracovní náklady při instalaci a konfiguraci honeypotu. Proto čím složitější a bohatší honeypot, tím je tento parametr významnější. Samozřejmě, čím složitěji a přesněji je nastaven systém - lákadlo, než je zapojeno služeb a parametrů, tím více je tento systém podobný skutečnosti.
- dle procesu používání a podpory
  Charakterizuje čas a úsilí při používání a podpoře honeypotu po procesu instalace a nastavení.
- dle sběru dat
  Charakterizuje množství dat, které honeypot může shromáždit o útočníkovi a jeho aktivitě. Podle úrovně interakce lze tyto prvky rozdělit na honeypoty s vysokou, střední a nízkou mírou interakce.

Tab. 2.1: Rozdělení honeypotů.

Míra interakce honeypotu	Úroveň simulace	Úroveň protokolování	Úroveň rizika	Proces instalace a konfigurace	Proces používání a podpory	Sběr dat
nízká	snadný	snadný	omezený	nízký	nízký	nízký
střední	střední	střední	variabilní	střední	střední	střední
vysoká	těžký	těžký	rozšířený	vysoký	vysoký	vysoký

Honeypot nízké míry interakce lze jednoduše nainstalovat, nakonfigurovat, používat a udržovat, protože mají jednoduchou strukturu a základní funkce. Tento honeypot napodobuje pouze část služeb. Útočník je omezen v interakci s těmito službami. Hlavním cílem honeypotu nízké míry interakce je detekce, funkce, vyšetření a neo-

právněné pokusy o připojení. Vzhledem k tomu, že tyto nástroje honeypotu poskytují omezenou funkcionalitu, většina z nich je prezentována jako programy. Program lze jednoduše nainstalovat na hostitele a přizpůsobit podle požadavků. Honeypot tohoto typu nelze použít k útoku nebo výzkumu jiných systémů. Řešení slabé interakce poskytují transakční informace. To jsou údaje shromážděné o okolnostech útoku, nikoliv však o útoku samotném. Pro honeypot nízké míry interakce je to následující:

- čas a datum útoku
- IP adresa a zdrojový port (útočník)
- IP adresa a cílový port (prostředky honeypot)

Honeypoty nízké míry interakce jsou zaměřeny na shromažďování známého chování. Útočník komunikuje obvyklým způsobem, a honeypot má předem určené druhy odpovědí. Honeypoty nízké míry interakce vykazují špatné, nebo dokonce negativní výsledky při interakci s neznámými nebo nepředvídatelnými druhy útoků. Honeypoty střední míry interakce mohou očekávat různou aktivitu a jsou navrženy tak, aby poskytovaly několik možných odpovědí na akci útočníka. Honeypot střední míry interakce vyžaduje trochu větší úsilí při instalaci a konfiguraci než slabé interakce. Obvykle nepřicházejí ve formě hotových softwarových řešení.

Využití a podpora honeypotu střední míry interakce je také složitější proces. Útočníci mají větší podíl na interakci se systémem, a proto je nutné provést další bezpečnostní opatření. Je třeba vyvinout mechanismy, které poskytnou jistotu, že útočník nebude schopen poškodit jiné systémy a zvýšená funkčnost nebude ohrožena. Kromě toho by měla být prováděna pravidelná podpora honeypoty, která spočívá v neustálém monitorování nových zranitelností.

Honeypot střední míry interakce je složitější a zvyšuje míru rizika. Zároveň je však schopen získat mnohem více informací. Na rozdíl od běžného skenování portů, je zde možnost záznamu aktivity virů, zkoumání stavu systému po útoku (například, jakým způsobem útočník pronikl do systému a zvýšil práva), ale také v získávání softwarových nástrojů útočníka.

Honeypot vysoké míry interakce poskytují větší množství informací o útočnících, ale vyžadují dostatek času na vybudování a podporu. Navíc přinášejí nejvyšší míru rizika. Cílem honeypotu silné interakce je poskytnout útočníkovi přístup k reálným operačním systémům, kde nic není, případně je simulováno nebo omezeno. Vybavení pro studium útoku je v tomto případě neuvěřitelné. Existuje možnost prozkoumat nové prostředky, odhalovat nové zranitelnosti v operačních systémech nebo aplikacích, ale také vědět, jak spolu útočníci interagují.

Ve většině případů jsou honeypoty silné míry interakce umístěny v kontrolovaném prostředí, např. v síti - po firewallu. Schopnost ovládat útočníka nepochází ze samotného honeypotu, ale z monitorovacího síťového zařízení - nejčastěji firewallu. Firewall

poskytuje útočníkovi možnost napadnout nástroj honeypot, ale zakazuje vytváření vnějších útoků. Vzhledem k tomu, že vybudovaná architektura je poměrně složitá, je nutné podrobně definovat rámec pravidel firewallu [13] [14] [15].

2.2 IoT HoneyPot

IoT honeypot nemůže být postaven na konvenční honeypot technologii. Různorodost vlastnosti zařízení internetu věcí je příčinou složitosti vývoje IoT honeypot a také velmi drahé a časově náročné. V současné době existuje několik IoT honeypotů, jejichž seznam je představen v tabulce 2.2 [16].

Tab. 2.2: Iot Honeypoty.

Honeypot	Míra interakce	Protokol, na který je zaměřený	Virtuální	Fyzický
Telnet IoTPOT	vysoká	Telnet	ano	ne
SIPHON	vysoká	SSH, HTTP	ne	ano
Multi-purpose IoT honeypot	vysoká	HTTP, SSH, TR-064, Telnet	ano	ne
Conpot	střední	Modbus (TCP), SNMP	ano	ne
IoTCandyJar	inteligentní	HTTP, SSH, Telnet, TR-064, XMPP, MQTT, UPnP, CoAP, MS-RDP	ano	ne
ThingPot	střední	HTTP, XMPP	ano	ne
HoneyThing	nízká	TR-064	ano	ne
ZigBee Honeypot	střední	ZigBee	ne	ano
Honeypot-camera	nízká	HTTP	ano	ne
MTPot	nízká	Telnet	ano	ne
Wificam	nízká	HTTP	ano	ne
HoneyThing	nízká	TR-069	ano	ne
Dionaea	nízká	FTP, HTTP, MSSQL, SIP, SMB, TFTP	ano	ne

Telnet IoT honeypot Hlavním účelem Telnet IoT honeypot je zachycení botnet útoků. Tento projekt zahrnuje server Python Telnet, který funguje jako honeypot pro malware IoT a dokáže automaticky analyzovat botnet spojení, mapové botnety a jejich sítě. Aplikace má architekturu klient/server, kde klient (skutečný honeypot)

přijímá telnetová připojení a server, který přijímá informace o připojeních a provádí analýzu. Backend server používá rozhraní HTTP, které slouží jak pro přístup k frontendu, tak i klientům k zasílání nových informací o připojení do backendu [17].

HoneyThing

Tento honeypot je určen pro TR-069 (CPE WAN Management Protocol). Je navržen tak, aby fungoval zcela jako modem/router s integrovaným webovým serverem RomPager a podporou protokolu TR-069 (CWMP) [18].

IoTPOT

Honeypot pro emulaci služeb Telnetu pro různá zařízení IoT. IoTPOT sestává z frontend lowinteraction respondentu spolupracujícího s backend highinteraction virtuálním prostředím zvaným IoTBOX. IoTBOX provozuje různá virtuální prostředí běžně používaná vestavěnými systémy pro různé CPU architektury [19].

Dionaea

Cílem Dionaea honeypotu je zachycení malware využívajícího zranitelnosti způsobené službami nabízenými do sítě, konečným cílem je získání kopie malwaru. Dionaea nabízí služby prostřednictvím tcp/udp a tls protokolu pro IPv4 a IPv6 a také používá omezení rychlosti a limity účtování pro připojení na TCP a TLS - pokud je to nutné [20].

ZigBee Honeypot

ZigBee Honeypot simuluje ZigBee bránu [21]. PPomocí honeytokenu je simulován provoz na ZigBee bráně, což je provedeno za účelem lákání útočníků. Tento honeypot sbírá data, když se útočník pokouší připojit na SSH port.

Komplexní honeypot, který emuluje platformu IoT v současné době neexistuje.

2.3 Mitigačni opatření

Správně implementovaný honeypot pomáhá v zabezpečení sítí a snižuje rizika. Při nesprávné implementaci však může honeypot dát útočníkovi další možnosti pro napadení. Při nasazování honeypotů musíme snížit šanci, že se něco stane s honeypotem samotným. Honeypoty by měly snižovat, nikoli zvyšovat riziko.

Prvním krokem ke snížení rizika je zaměření na úroveň interakce. Čím vyšší je úroveň interakce, tím větší je složitost a riziko, že se něco pokazí [14].

Jail nebo chroot prostředí může být použito ke shromažďování více informací o útočníkovi, protože tato řešení dávají útočníkovi možnost interakce s operačním systémem. Zkušený útočník však může být schopen prolomit takové kontrolované prostředí. Může být objeveno nové zneužití (exploit), vyvinuta nová metoda, může se vyskytnout chyba v implementaci jail prostředí nebo může dojít k chybě. Různé známé a neznámé věci se mohou pokazit a zvětšit riziko zneužití honeypotu. Aby se

toto riziko snížilo, při vytváření honeypotu je důležité dosáhnout tak malou interakci, jaká je nezbytná pro cíl který má být dosažen honeypotem. Snížením úrovně interakce lze snížit složitost i riziko.

Honeypoty s nízkou interakcí samy o sobě nabízejí malé riziko, protože pouze emulují jiné služby nebo operační systémy. Musíme však zajistit, aby byl základní operační systém bezpečný. Náš honeypot software může být bezpečný, ale zranitelný operační systém může být snadno ohrožen. Před instalací honeypotu musíte dodržovat doporučené postupy pro bezpečný operační systém. Mezi příklady patří vypnutí jakékoli služby, kterou nepotřebujete, a oprava všech služeb, které jsou používány. Každý operační systém má jedinečné požadavky na zabezpečení. Je důležité tyto požadavky přečíst a porozumět jim před nasazením honeypotů s nízkou interakcí [14].

Předpokládá se, že operační systémy honeypotů s vysokou interakcí mohou být ohroženy. Musí být použit jiný mechanismus ke zmírnění rizika. V těchto situacích je potřeba implementovat řízení dat. Řízení dat se používá k omezení příchozí a odchozí aktivity útočníka. Cílem je zajistit, aby útočník, který má kontrolu nad honeypotem, nemohl nadále použít honeypot k útoku nebo poškození jiných nonhoneypotových systémů. Jednoduchým způsobem, jak dosáhnout tohoto cíle, je umístit firewall před honeypot, který umožní jakékoli příchozí připojení, ale zablokuje všechny pokusy o odchozí připojení. U sofistikovanějších honeypotů lze implementovat pokročilé mechanismy kontroly dat. Čím jednodušší mechanismus kontroly dat je použit, tím menší je riziko, že dojde k chybě.

U honeypotu se střední mírou interakcí je potřeba používat nějakou kombinaci zabezpečení operačního systému a implementace řízení dat. Vybírání metod závisí na typu honeypotu střední interakce. Například pokud bude vytvářeno jail prostředí v systému Unix, je potřeba použít oba mechanismy pro snížení rizika. To by pomohlo ovládnout útočníka a zabránit mu v tom, aby prolomil jail prostředí a dostal se k reálnému OS. Poté je možné nasadit mechanismus kontroly dat, například firewall. Firewall zastaví útočníka v případě, že prolomí jail prostředí.

Dalším krokem ke snížení rizika je mechanismus testování. Je potřeba otestovat honeypot před nasazením. Pokud se honeypot nebude chovat tak, jak je očekáváno, a nebude o tomto selhání informovat, může honeypot představovat větší riziko, než zabezpečení. Může se stát, že jedna z emulovaných služeb přestane fungovat a nedokáže detekovat útoky. Nebo se stane, že varovný mechanismus je nesprávně nakonfigurován a upozornění zasílá e-mailem na nesprávnou e-mailovou adresu. Při útoku na honeypot může být upozorněna nesprávná osoba nebo neexistující adresa, zatímco správce o útoku nebude vědět [14].

2.4 Metody odhalování honeypotu

Jakmile je honeypot identifikován, ztrácí svou hodnotu. Pokud například útočník objeví honeypot, který se používá k detekci útoků, dozví se, že to není reálný systém a nebude na něj útočit. Ještě horší je, že může toto zjištění sdělit dalším útočníkům, kteří se tímto také dozvědí o existujícím honeypotu. Níže jsou uvedeny metody detekce honeypotu.

Známe chování honeypotů

Úprava chování honeypotů se týká hlavně komerčních nebo předbalených řešení honeypotů. Jakékoli řešení honeypot, které si můžete stáhnout a nainstalovat, už je potenciálním útočníkům známo. Pokud si stáhnete nebo zakoupíte řešení honeypot a použijete jej pro svou organizaci, není známé, že útočník nemohl získat přístup a naučit se stejné technologii. Taková řešení mohou mít standardní konfigurace nebo specifické chování, které lze identifikovat. Jakmile jsou tyto podpisy rozpoznány, honeypot je identifikovaný. Výchozí instalace honeypot může například přijít s deseti emulovanými službami. Útočník může identifikovat těchto deset služeb, verzi každé emulované služby a identifikovat honeypot řešení. Klíčem k vyhnutí se tomuto typu detekce je změna funkčnosti a identity řešení honeypot, kterému čelíte. Po instalaci honeypotu je potřeba upravit výchozí nastavení. Zakázat některé služby, které nejsou potřebné, a přidat nějaké další služby, které naopak potřebné jsou. Cílem je vytvořit jedinečné vlastnosti, které ztíží identifikaci honeypotu. Vlastnoručně vytvořené honeypoty nemají tento problém, protože jsou každý jedinečný.

Hodně otevřených portů

Když útočník kontroluje IP adresy, jestli mají zranitelnosti nebo ne, například spouští nmap kontrolu, jestli mají otevřené porty nebo ne. Když bude hodně otevřených portů jako výsledek spouštění nmap, znamená to, že se pravděpodobně jedná o honeypot, jelikož moderní server s firewallem by nikdy nezobrazil takový výsledek.

Zpoždění

Podstatné detekční riziko pro honeypoty pramení ze skutečnosti, že zpoždění jsou zaváděna do procesů, které napodobuje honeypot (např. Autentizace přes SSH, která je zpožděna kvůli dalšímu protokolování nebo předávání logických údajů, které vyžaduje honeypot), které by jinak takové zpoždění nepozorovalo. Tato zpoždění umožňují útočníkům detekovat honeypot.

Zastaralý OS

Při použití netcat se zobrazí OS příslušné IP, a pokud to bude zastaralá verze, pravděpodobně je to honeypot.

Špatná konfigurace

Honeypoty lze snadno detekovat, pokud se nechovají jako skutečné systémy. To platí zejména o emulovaných honeypotech. Například honeypot může emulovat operační

systém Solaris, ale skutečný operační systém honeypot je založen na Windows. Když se útočník připojí k emulované službě, chová se honeypot jako server Solaris. Pokud však útočník použije pokročilé techniky detekce operačního systému, jako je například analýza zásobníku IP honeypotu, může útočník systém identifikovat jako operační systém Windows. Tyto nesrovnalosti v typech operačních systémů lze použít k identifikaci honeypotu. Tyto problémy se týkají zejména honeypotů, které emulují služby.

3 Praktické řešení

3.1 Návrh a sestavení experimentálního prostředí

Pro sestavení vlastní LoRaWAN sítě byla použita následující zařízení:

1. LoRaWAN brána
   Brána sestává z IC880A SPI koncentrátoru, raspberry Pi 3 B+, redukční desky IC880A, napájecího zdroje, pigtail pro iC880A-SPI, antény 868MHz a antény 868MHz.
2. Koncová zařízení
   Smart Plug a senzor teploty značky nke Watteco [22] [23].
   Smart Plug je inteligentní domácí zařízení LoRaWAN třídy C, které monitoruje, ovládá, reportuje, připojuje domácí elektroniku a kvalitu elektrického vedení. Senzor teploty a vlhkosti je bezdrátové senzorové zařízení LoRaWAN s vlastním pohonem, které měří a přenáší teplotu a vlhkost na velké vzdálenosti.
3. RTL-SDR DVB Tuner splňující funkci přijímače.
4. Virtual Machine Ubuntu na které byly nainstalovány aplikační a síťový server.

Prvním krokem byla instalace Raspbian na SD-card. Podrobné kroky jsou zobrazeny níže.

```
list disk
select disk 1
clean
create partition primary
select partition 1
active
format FS=NTFS label=Data quick
```

Pro zajištění vzdáleného přístupu k bráně bylo potřeba vytvořit “ssh” soubor na instalační SD kartu. Při prvním spuštění Raspberry Pi bylo povoleno SPI, SSH, dále bylo povoleno rozbalení souborového systému (expand the filesystem). Pro bezpečnostní účely byl vytvořen nový user ttn s právem superusera, defaultní pi user byl odstraněn. Kroky popsané výše byly uskutečněny pomocí následujících příkazů:

```
$ sudo raspi-config
$ sudo adduser ttn
$ sudo adduser ttn sudo
$ sudo visudo
```

Po tomto příkazu bylo potřeba přidat řádek

```
ttn ALL=(ALL)  NOPASSWD: ALL
```

```
$ sudo userdel -rf pi
```

Pro konfiguraci brány byl použit git repozitář [24], po jehož stažení byl spuštěn instalační skript. To bylo uskutečněno pomocí následujících příkazů

```
$ git clone https://github.com
ttn-zh/ic880a-gateway.git~/ic880a-gateway
$ cd ~/ic880a-gateway
$ sudo ./install.sh
```

Během instalace bylo zobrazeno unikátní EUI číslo brány. Běžící brána je zobrazena na obr. 3.1.

```
● ttn-gateway.service - The Things Network Gateway
   Loaded: loaded (/lib/systemd/system/ttn-gateway.service; enabled; vendor preset: enabled)
   Active: active (running) since Thu 2020-06-04 14:22:35 CEST; 1min 50s ago
 Main PID: 976 (start.sh)
    Tasks: 4 (limit: 2200)
   Memory: 900.0K
   CGroup: /system.slice/ttn-gateway.service
           ├─976 /bin/bash /opt/ttn-gateway/bin/start.sh
           └─998 ./poly_pkt_fwd

Jun 04 14:22:35 ttn-gateway ttn-gateway[976]: INFO: redefined parameters will overwrite global parameters
Jun 04 14:22:35 ttn-gateway ttn-gateway[976]: INFO: local_conf.json does not contain a JSON object named SX1301_conf
Jun 04 14:22:35 ttn-gateway ttn-gateway[976]: INFO: local_conf.json does contain a JSON object named gateway_conf, parsing
Jun 04 14:22:35 ttn-gateway ttn-gateway[976]: INFO: gateway MAC address is configured to B827EBFFFE859096
Jun 04 14:22:35 ttn-gateway ttn-gateway[976]: INFO: Found 1 servers in array.
```

Obr. 3.1: Běžící brána

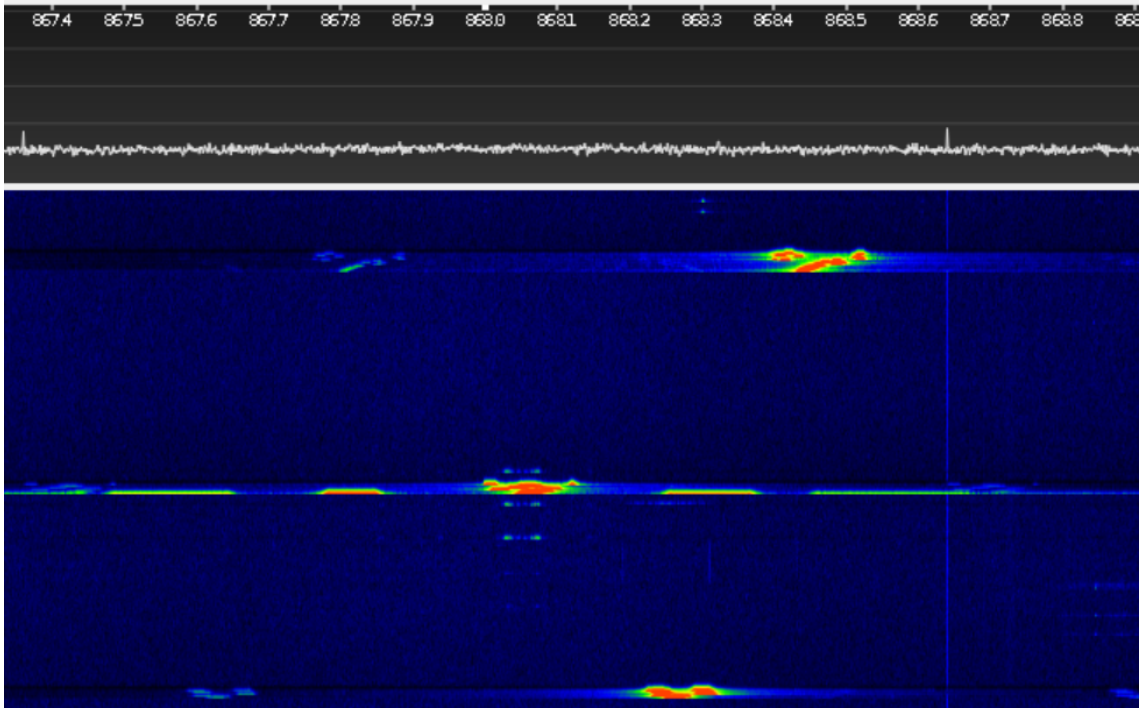
Během konfigurace brány byl taky nainstalován packet-forwarder. Packet-forwarder je program, který běží na bráně, a předává pakety z koncentrátoru na server přes IP/UDP spojení, stejně jako pakety, které jsou odesílány ze serveru. V konfiguračních souborech packet-forwarderu local_conf.json a global_conf.json bylo potřeba změnit EUI brány a také adresu serveru, se kterým bude probíhat komunikace. Následujícím krokem byla registrace brány na `https://www.thethingsnetwork.org/` Po úspěšné registraci brány bylo možné pozorovat provoz na bráně a zprávy, kterými komunikovaly brána a koncová zařízení mezi sebou (obr. 3.2).

▲ 17:13:07	868.1	lora	4/5	SF 11 BW 125	577.5	1	dev addr: 5E 00 0F 13	payload size: 12 bytes
▲ 17:13:01	868.5	lora	4/5	SF 11 BW 125	577.5	1	dev addr: 5E 00 0F 13	payload size: 12 bytes
▲ 17:12:54	868.1	lora	4/5	SF 10 BW 125	288.8	1	dev addr: 5E 00 0F 13	payload size: 12 bytes
▲ 17:12:48	868.3	lora	4/5	SF 10 BW 125	288.8	1	dev addr: 5E 00 0F 13	payload size: 12 bytes

Obr. 3.2: Provoz na bráně

Za pomoci zařízení RTL-SDR DVB Tuner a aplikace CubicSDR byla komunikace odchycena (obr. 3.3). Po úspěšném zprovoznění zařízení byl proveden odposlech na

frekvenci 868 MHz, na obrázku níže lze zjistit, na jaké frekvenci probíhala komunikace a na kterých kanálech. V tomto případě na kanálech 1, 3 a 5.



Obr. 3.3: Zachycená komunikace v aplikace CubicSDR

Po úspěšném připojení brány na evropský server router.eu.thethings.network byl vytvořen vlastní síťový a aplikační server pomocí open-source řešení Chirpstack Server [25]. Byl použit software Ubuntu 18.04 nainstalovaný na virtuálním přístroji. Prerekvizitou pro instalaci aplikačního a síťového serveru byl běžící MQTT server a PostgreSQL, jelikož servery zapisují data do databáze. Prerekvizity byly nainstalovány tímto příkazem. Síťový server pro svou funkci potřebuje také redis server.

```
sudo apt-get install mosquitto
sudo apt-get install postgresql
sudo apt-get install redis-server
```

Před instalací aplikačního serveru bylo potřeba vytvořit uživatele a databázi:

```
sudo -u postgres psql
-- create the chirpstack_as user
create role chirpstack_as with login password 'dbpassword';
-- create the chirpstack_as database
create database chirpstack_as with owner chirpstack_as;
-- enable the trigram and hstore extensions
```

```
\c chirpstack_as
create extension pg_trgm;
create extension hstore;
-- exit the prompt
\q
```

Pro aktivaci Chirpstack repozitáře byly použity následující příkazy:

```
sudo apt-key adv --keyserver keyserver.ubuntu.com
--recv-keys 1CE2AFD36DBCCA00
sudo echo deb https://artifacts.chirpstack.io/packages
/3.x/deb stable main| sudo tee /etc/apt/sources.list.d/
chirpstack.list
sudo apt-get update
```

Po aktivaci repozitáře lze spustit instalaci síťového serveru pomocí příkazu:

```
sudo apt-get install chirpstack-application-server
```

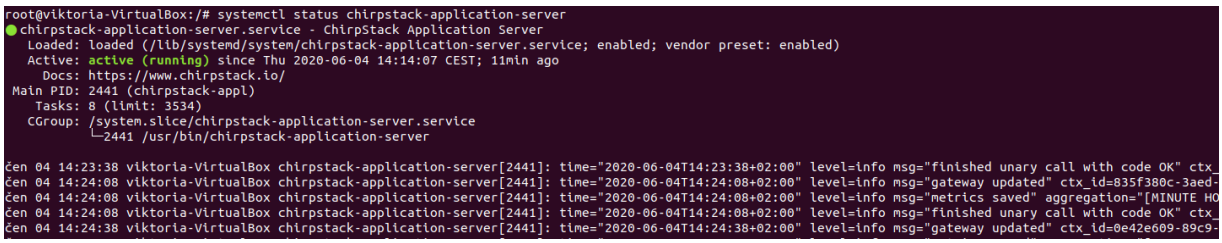
Následujícím příkazem lze nastartovat, zastavit a zobrazit status serveru.

```
sudo systemctl [start|stop|restart|status]
chirpstack-application-server
```

Patřičné logy lze zobrazit pomocí příkazu:

```
journalctl -u chirpstack-application-server -f -n 50
```

Při sledování statusu aplikačního serveru byla nalezena chyba. Chyba byla způsobena tím, že v konfiguračním souboru chirpstack-application-server.json nebyly správně nastaveny hodnoty dsn a jwt_secret. Po změně těchto parametrů aplikační server běžel bez chyb (obr. 3.4).



Obr. 3.4: Status aplikačního serveru bez chyb

Před instalací síťového serveru byla potřeba vytvořit uživatele a databázi:

```
sudo -u postgres psql
create role chirpstack_ns with login password 'dbpassword';
```

```
create database chirpstack_ns with owner chirpstack_ns;
\q
```

Pro aktivaci Chirpstack repozitáře byly použity následující příkazy:

```
sudo apt-key adv --keyserver keyserver.ubuntu.com
--recv-keys 1CE2AFD36DBCCA00
sudo echo deb https://artifacts.chirpstack.io/packages
/3.x/deb stable main |sudo tee /etc/apt/sources.list.d/
chirpstack.list
sudo apt-get update
```

Po aktivaci repozitáře lze spustit instalaci síťového serveru pomocí příkazu:

```
sudo apt-get install chirpstack-network-server
```

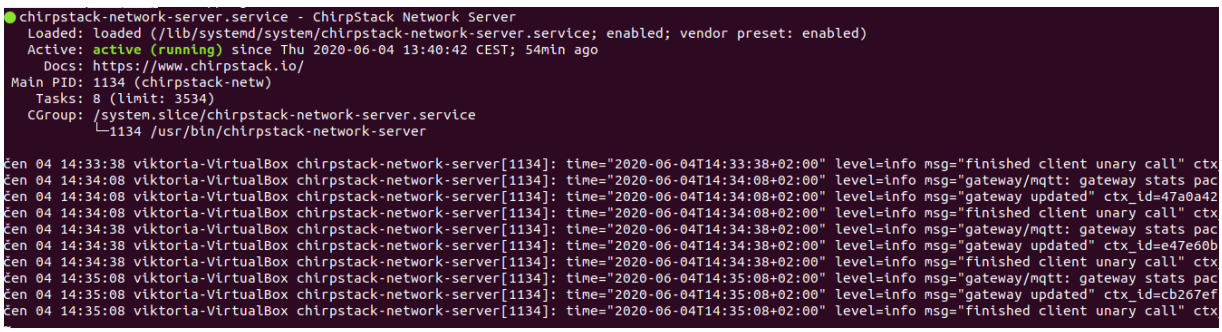
V konfiguračním souboru chirpstack-network-server.json nebyla správně nastavena hodnota dsn. Bylo potřeba změnit tento parametr pro zajištění správné funkce síťového serveru. Následujícím příkazem lze nastartovat, zastavit a zobrazit status serveru.

```
sudo systemctl [start|stop|restart|status]
chirpstack-network-server
```

Patřičné logy lze zobrazit pomocí příkazu

```
journalctl -u chirpstack-network-server -f -n 50
```

Po provedení kroků popsaných výše síťový server běžel bez chyb (obr. 3.5).



Obr. 3.5: Status síťového serveru bez chyb

Pro komunikaci se servery je potřeba nainstalovat na bráně službu, která transformuje obsah UDP paketů z programu packet-forwarder na JSON zprávy do MQTT protokolu. Tato služba se nazývá gateway-bridge. Pro aktivaci Chirpstack repozitáře byly použity následující příkazy:

```
sudo apt-key adv --keyserver keyserver.ubuntu.com
--recv-keys 1CE2AFD36DBCCA00
sudo echo "deb␣https://artifacts.chirpstack.io/packages/3.x/
deb␣stable␣main" | sudo tee /etc/apt/sources.list.d/
chirpstack.list
sudo apt-get update
```

Následovně proběhla instalace gateway-bridge:

```
sudo apt-get install chirpstack-gateway-bridge
```

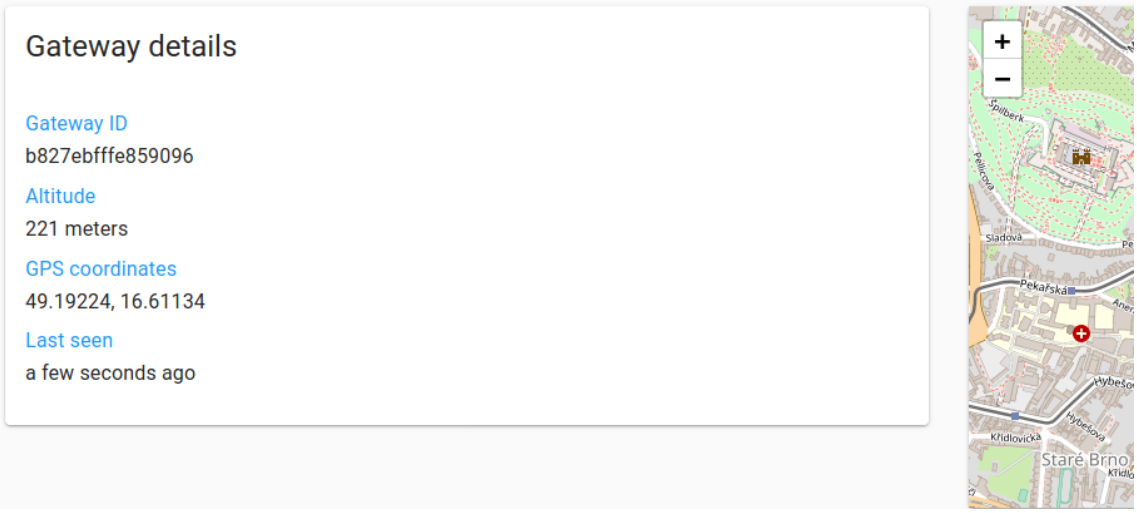
Následujícím příkazem lze nastartovat, zastavit, zobrazit status server.

```
sudo systemctl [start|stop|restart|status]
chirpstack-gateway-bridge
```

Patřičné logy lze zobrazit pomocí příkazu:

```
journalctl -u chirpstack-gateway-bridge -f -n 50
```

Po nastavení výše popsaných jednotek síť fungovala v pořádku: brána navázala spojení se síťovým serverem (obr. 3.6).



Obr. 3.6: Navázané spojení brány se síťovým serverem

3.2 Sestavení honeypotu

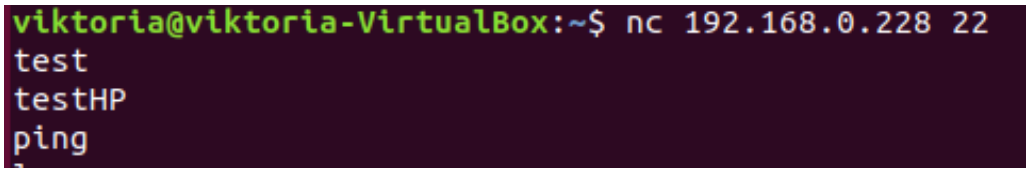
Pro vytváření honeypotu je použit hardware, na kterém byla postavena brána. Na raspberry pi byl nainstalován BerryBoot, s jehož pomocí byly vytvářeny 2 obrazy disků: jeden pro honeypot nízké míry interakce, druhý pro honeypot vysoké míry

interakce. Honeypot nízké míry interakce simuluje službu do přihlášení. Pro realizace honeypotu nízké míry interakce byl napsán script v bash:

```
!/bin/bash
iptables -t nat -A PREROUTING -p tcp -dport 22 -j REDIRECT
-to-ports 10000 &
netcat -l -p 10000 | tee -a /home/Cat.log &
tcpdump -i wlan0 -w /home/tcpdump.pcap &
echo HP is running
```

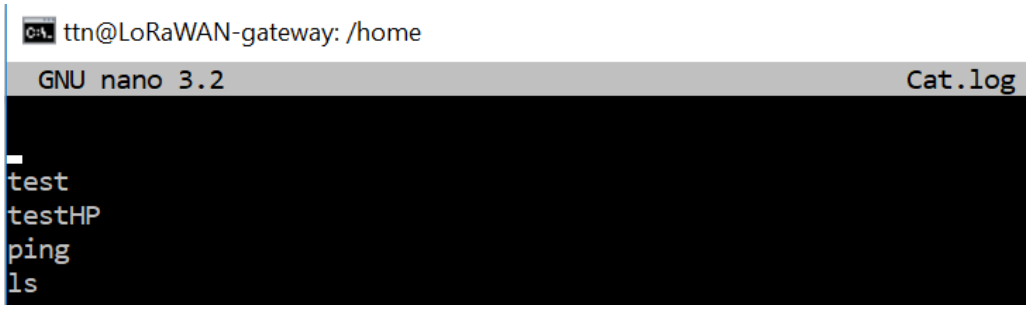
Kde:
iptables: administrační nástroj pro filtrování paketů IPv4 a NAT
-t: tato volba určuje tabulku shody paketů, se kterou by příkaz měl pracovat. Pokud je jádro konfigurováno s automatickým načítáním modulů, bude proveden pokus o načtení vhodného modulu pro tuto tabulku, pokud tam již není.
nat: tato tabulka je konzultována, když dojde k paketu, který vytváří nové připojení. To sestává ze tří vestavěných: PREROUTING (pro změnu paketů, jakmile přijdou), OUTPUT (pro změnu lokálně generovaných paketů před směřováním) a POSTROUTING (pro změnu paketů, které se chystají jít ven).
-A PREROUTING: připojí jedno nebo více pravidel na konec vybraného řetězce "PREROUTING". Pokud se názvy překládají na více než jednu adresu, přidá se pravidlo pro každou možnou adresu kombinace.
-p tcp: protokol pravidla nebo paketů ke kontrole. Zadaný protokol může být protokol tcp, udp, udplite, icmp, esp, ah, sctp nebo speciální klíčové slovo „all", nebo to může být číselná hodnota, představující jeden z těchto protokolů nebo jiný. „all" se bude shodovat se všemi protokoly a bude vynecháno jako výchozí, pokud je tato možnost vynechána.
-j REDIRECT: určuje cíl pravidla; tj. co dělat, když se s ním paket shoduje. V tomto případě přesměrují připojení z portu 22 (SSH port) na port 10000.
netcat: příkaz nastavuje TCP spojení s označeným uzlem a číslem portu. V zásadě se jedná o analogický starý příkaz telnet v systému Linux.
-l: režim poslechu
-p 10000: číslo portu, na kterém se odposlouchává (v tomto případě port 10000). Používá se s volbou -l
tee: čtení ze standardního vstupu a zápis na standardní výstup a soubory
-a připojí se k souborům, nepřepisuje je
tcpdump: výpis provozu na síti
-i wlan0: poslouchá na rozhraní. Pokud není uvedeno, tcpdump prohledá v seznamu systémových rozhraní nejnižší číslované konfigurované rozhraní (kromě zpětné smyčky).
-w: zapisuje nezpracované pakety do souboru.

echo: vypisuje zprávu.
Na obrázku 3.7 jsou zobrazené zprávy poslané na brány.



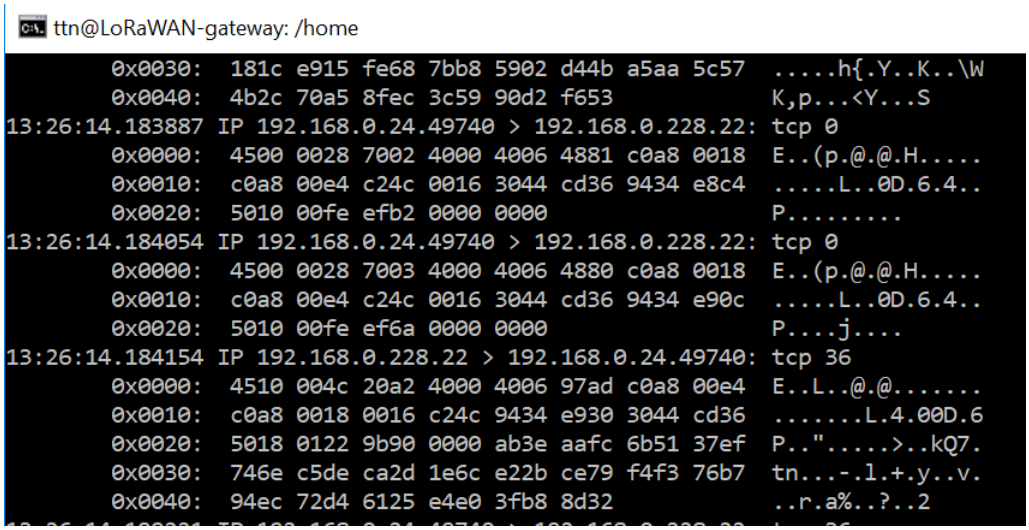
Obr. 3.7: Zprávy poslané na bránu

Na obrázku 3.8 je zobrazen obsah souboru Cat.log



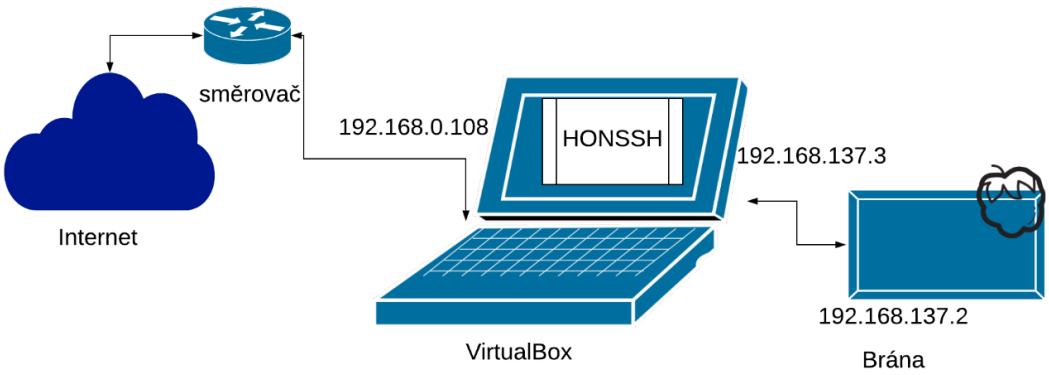
Obr. 3.8: Zachycené zprávy na bráně

Na obrázku 3.10 je zobrazen obsah souboru tcpdump.pcap



Obr. 3.9: Zachycená komunikace mezi bránou a "útočníkem"

Pro vytváření honeypotu vysoké míry interakce byl použit hardware, na kterém je postavena brána a taky virtuální stroj Ubuntu a HONSSH open-source řešení [26]. Byla sestavena topologie (obr. 3.10).



Obr. 3.10: Topologie sítí

"Útočník" se bude připojovat na IP 192.168.0.228, následovně bude přesměrován na 192.168.137.2 přes 192.168.137.3 IP – adresu. Celá jeho aktivita bude zaznamenána do logu na HONSSH stroje. HONSSH komunikují s Internetem přes enp0s3 rozhraní. S bránou HONSSH komunikuje pomocí kabelu Ethernet.
HONSSH instalace: V prvním kroku pomocí příkazu

```
sudo nano /etc/ssh/sshd_config
```

byly provedeny úpravy v konfiguračním souboru SSH. Port byl změněn z 22 na 2222 a také bylo zakázáno přihlášení přes SSH pomocí hesla – pro bezpečnostní účely. Následujícím příkazem byly nainstalovány potřebné balíčky:

```
apt-get install git python python-dev python-pip
libffi-dev libssl- dev libgeoip-dev mysql-client
```

a virtuální prostředí:

```
pip install virtualenv
```

Pomocí příkazu

```
mkdir /home/honssh
```

byl vytvořen adresář pro umístění HONSSH. Jelikož je pro zajištění bezpečnosti potřeba spouštět HONSSH script useru, který nemá root práva, byl vytvořen honssh user:

```
useradd honssh
chown honssh:honssh /home/honssh
```

Následovně je potřeba zkopírovat HONSSH repozitář s gitu:

Pomocí příkazu níže bylo provedeno přepnutí do virtuálního prostředí a nainstalovány potřebné balíčky:

```
virtualenv honssh_env
source ~/honssh_env/bin/activate
pip install -r ~/honssh/requirements
```

Pomocí následujících příkazů byly zkopírovány konfigurační soubory:

```
cp ~/honssh/honssh.cfg.default ~/honssh/honssh.cfg
cp ~/honssh/users.cfg.default ~/honssh/users.cfg
```

Byla upravená konfigurace HONSSH pomocí příkazu:

```
nano ~/honssh/honssh.cfg
```

ssh_addr: je potřeba nastavit na IP adresu HonSSH připojeném k „Internetu“ (192.168.0.228).
ssh_port: port, na kterém by měl HonSSH poslouchat příchozí spojení SSH (2220).
client_addr: IP adresa HonSSH připojena k bráně (192.168.137.2).
sensor_name:název HONSSH v tomto případě “HonsshHP“.
honey_ip: Nastavení IP adresy honeypotu (192.168.137.2).
Protože HonSSH bude také fungovat jako směrovač pro honeypot, je potřeba nastavit přesměrování IP. Pomocí následujícího příkazu bylo povoleno přesměrování IP:

```
sudo sysctl net.ipv4.ip_forward=1
```

Pomocí nastavení pravidel iptables byla zajištěna bezpečnost HONSSH:

```
iptables -N LOGDROP
```

vytvoří nový řetězec LOGDROP, který loguje pakety do souboru a po záznamu pak pakety zahodí.

```
iptables -A INPUT -m conntrack --ctstate INVALID -j DROP
```

přidává pravidlo na konec řetězce INPUT, které zahodí paket, pokud není spojen s žádným známým připojením

```
iptables -A INPUT -m conntrack --ctstate ESTABLISHED,
RELATED -j ACCEPT
```

přidává pravidlo na konec řetězce INPUT, které povolí paket, pokud souvisí s připojením, kterým pakety již prošly oběma směry (ESTABLISHED) anebo paket iniciuje nové připojení, ale je také spojen s existujícím připojením, například při přenosu dat přes FTP nebo chybovou zprávu ICMP(RELATED)

```
iptables -A INPUT -i enp0s8 -j ACCEPT
```

povolí příchozí paket na rozhraní enp0s8(192.168.0.108).

```
iptables -A INPUT -i enp0s3 -j LOGDROP
```

bude logovat příchozí paket na rozhraní enp0s3(192.168.137.3).

```
iptables -P INPUT DROP
```

Zakáže všechny příchozí komunikace kromě povolených výše.

```
iptables -A LOGDROP -j LOG --log-prefix "fw␣děný:␣"
```

Přidá pravidlo k řetězcům LOGDROP, podle kterého se před logem bude označovat: "fw deny: "

```
iptables -A LOGDROP -j DROP
```

Zahodí všechny pakety shodující se s LOGDROP řetězcem.
Také byly nastaveny iptables na bráně:

```
iptables -A FORWARD -m conntrack --ctstate INVALID -j DROP
```

přidává pravidlo na konec řetězce FORWARD, které zahodí paket, pokud není spojen s žádným známým připojením

```
iptables -A FORWARD -m conntrack --ctstate ESTABLISHED,
RELATED -m limit --limit 100/s -j ACCEPT
```

přidává pravidlo na konec řetězce FORWARD, které povolí paket, pokud souvisí s připojením, kterým pakety již prošly oběma směry (ESTABLISHED) anebo paket iniciuje nové připojení, ale je také spojen s existujícím připojením, například při přenosu dat přes FTP nebo chybovou zprávu ICMP(RELATED). Pravidlo používající rozšíření – limit se bude implementovat do dosažení tohoto limitu.

```
iptables -A FORWARD -i eth0 -o enp0s3 -j ACCEPT
```

Povolí pakety odeslané s enp0s3 rozhraní HONSSH na eth0 rozhraní na bráně.

```
iptables -A FORWARD -d 192.168.0.0/16 -j LOGDROP
```

Přidá pravidlo k řetězcům FORWARD, podle kterého pro všechny přeposlané pakety z 192.168.0.0/16 bude použit LOGDROP.

```
iptables -A FORWARD -p tcp --dport 22 -j DROP
```

Zahodí přeposlané pakety na 22 port (SSH).

```
iptables -A FORWARD -m limit --limit 1/s -j ACCEPT
```

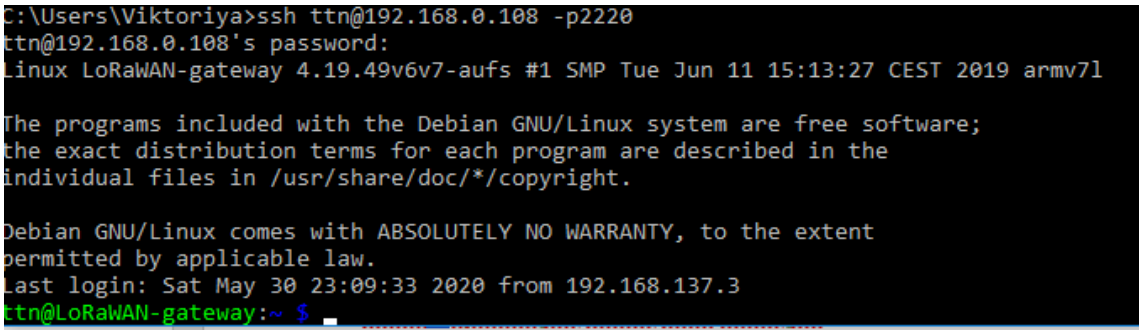
Přidá pravidlo k řetězci FORWARD, které povolí všechny příchozí komunikace, pokud nebude dosaženo limitu 1/s.

```
iptables -P FORWARD DROP
```

Přidá pravidlo k FORWARD řetězci, které zahodí všechny pakety kromě povolených nahoře. Pomocí příkazu

```
twistd -y honssh.tac -p honssh.pid
```

byl nastartován honeypot. Níže je uvedený scénář útoku: předpokládá se, že user ttn má slabé heslo a útočník ho odhalil a dostal se na bránu. První jsem se připojila na honeypot pomoci ssh, jak je zobrazeno na obrázku 3.11

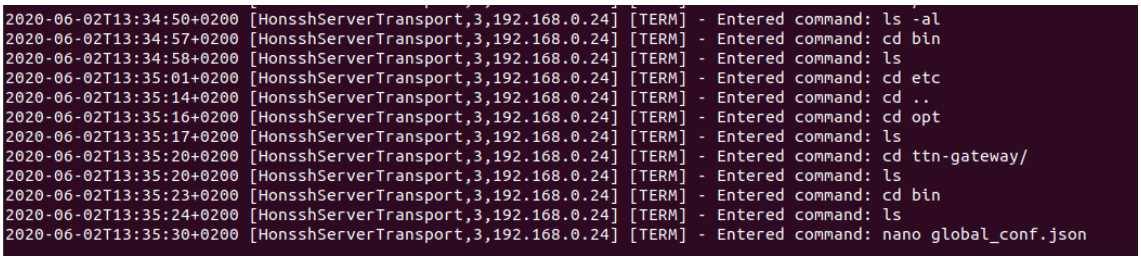


Obr. 3.11: Přihlášení na bránu.

Níže na obrázku 3.12 jsou uvedeny screenshoty příkazů které zadával "útočník"na bráně.

Obr. 3.12: Příkazy zadávané „útočníkem“.

Následovně na obrázku 3.13 je zobrazen výpis logu z HONSSH, který zaznamenával každý krok provedený na bráně.



Obr. 3.13: Výpis logu HONSSH.

Tím pádem je vytvořený funkční high-interaction honeypot a vytvořena topologie sítě, která zabezpečuje i HONSSH server, i brány (honeypot). Slabinou tohoto řešení je to, že „útočník “ může eskalovat práva na root uživatele a tím získá neomezený přístup k bráně. HONSSH taky umožňuje nastavení přihlašovacích údaje pomocí souboru /honssh/users.cfg. Pomocí toho souboru se dá nastavit falešné heslo pro přístup na bránu, když reálné heslo bude uloženo v konfiguračním souboru na HONSSH, a „útočník“ se o něm nikdy nedozví, ale dostane se na bránu pomocí lehce odhalitelného hesla. Také se dá nakonfigurovat, že přihlašování s určitým loginem a heslem projde jen s nějakou pravděpodobností. Nedostatkem tohoto je, že „útočník“ může snadno odhalit honeypot, proto jsem použila řešení popsané výše. Toto řešení lze použít, pokud na bráně nejsou umístěna citlivá data. V případě umístění citlivých dat na bráně je možnost použít chroot/jail řešení.
Chroot/jail je způsob izolace procesů od zbytku systému. Měl by být používán pouze pro procesy, které neběží jako root, protože uživatelé root se mohou velmi snadno vymanit do reálného procesu. Je to realizované tím, že se vytváří adresářový strom, ve kterém jsou zkopírovány nebo propojeny všechny systémové soubory potřebné pro spuštění procesu. Pomocí systémového volání chroot () se mění kořenový adresář, který bude na základně tohoto nového stromu, a zahájí se proces spuštěný v tomto prostředí chroot. Protože ve skutečnosti nemůžeme odkazovat cesty mimo upravený kořenový adresář, nelze na těchto místech provádět škodlivé operace (čtení/ zápis atd.).
Bylo potřeba vytvořit složku, ve které se umístí jail pomocí příkazu mkdir, a nastavit práva pomocí příkazu chmod.

```
mkdir -p /home/jail
chown root:root /home/jail
chmod 0755 /home/jail
```

Ve složce jail byly vytvořeny následující složky odpovídající hlavnímu kořenovému adresáři:

```
sudo mkdir usr
```

```
sudo mkdir bin
sudo mkdir etc
sudo mkdir lib
sudo mkdir tmp
sudo mkdir var
sudo mkdir opt
sudo mkdir sbin
```

Následovně byla vytvořena složka dev a v ní pomocí příkazu mknod vytvořeny speciální soubory. mknod - vytvoří speciální soubor; -m nastaví bitová práva na 666 pro soubor (jako chmod) první číslo-major druhé – minor:

```
# mkdir -p /home/jail/dev/
# cd /home/jail/dev/
# mknod -m 666 null c 1 3
# mknod -m 666 tty c 5 0
# mknod -m 666 zero c 1 5
# mknod -m 666 random c 1 8
```

Pomocí příkazu

```
sudo cp /etc/command -R /home/jail/etc
```

byly nakopírovány následující příkazy do složky etc v jail prostředí: bash, ls, cp, chmod, dir, echo, ip, hostname, kill, less, login, mkdir, moře, mv, nano, ping, ps, pwd, rm, sh, sleep, rmdir, su, touch, uname.
Pomocí příkazu

```
sudo ldd /etc/prikaz
```

byly zjištěny knihovny, které obsahují každý z příkazů a byly nakopírovány do složek /home/jail/usr/lib; /home/jail/lib;. Obsah složky /etc byl celý zkopírován pomocí příkazu, následně byly soubory s citlivými daty upraveny.

```
cp -avr /etc /home/jail/etc
```

V následujícím kroku byl vytvořen uživatel toor pomocí příkazu useradd a nastaveno heslo pomocí příkazu passwd:

```
useradd toor
passwd toor
```

Do jail byly vykopírovány následující soubory, protože se v nich přidaly data ohledně chroot uživatele:

```
cp -vf /etc/{passwd,group} /home/test/etc/
```

Pomocí příkazu

```
náno /etc/ssh/sshd_config
```

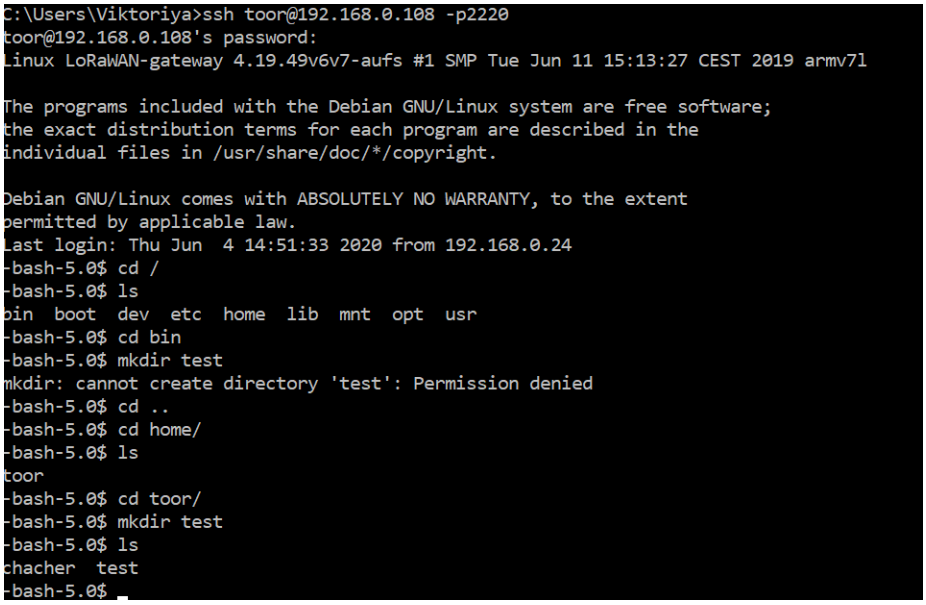
byl otevřen sshd_config soubor a upraveny následující řádky:

```
Match User toor
#specify chroot jail
ChrootDirectory /home/jail
```

Pomocí úprav v souboru sshd_config bylo zajištěno, že při připojení přes SSH bude user přesměrován do jailu. Následovně byla potřeba restartovat sshd službu pomocí příkazu systemctl:

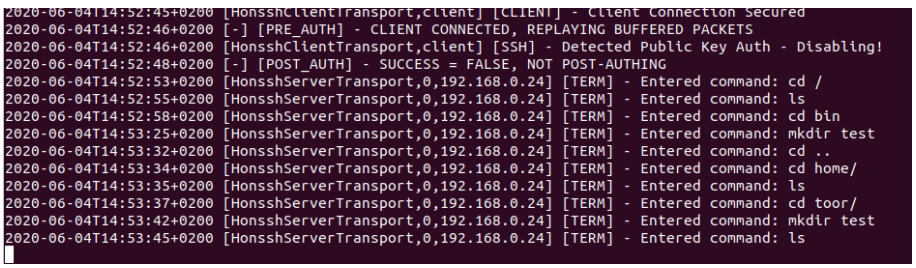
```
systemctl restart sshd
```

Po otestování vše funguje správně, po připojení přes SSH se user přesměroval do změněného kořenového adresáře, to lze vidět na obrázku 3.14



Obr. 3.14: Připojení na bránu useru toor.

HONSSH zaznamenává jakoukoliv aktivitu na bráně (obr. 3.15).



Obr. 3.15: Výpis logu HONSSH jail/chroot.

Závěr

V teoretické části bakalářské práce byla řešena problematika LoRaWAN. Byla popsána komunikace prvku v síti LoRaWAN a provedena analýzu útoků na IoT zařízení. Analýza byla zaměřena především na bezpečnost v rámci LoRaWAN. Byly popsány útoky a mitigační opatření. Dále se práce zabývala popisem honeypotů, uvedením jejich výhod a nevýhod. Následně byla popsána kritéria pro rozdělení honeypotů a prozkoumány IoT honeypoty současné doby. Komplexní honeypot, který emuluje platformu IoT v současné době neexistuje.

V praktické části byla sestavena vlastní LoRaWAN síť, zkonstruovány a nakonfigurovány jednotlivé součásti této sítě. V této síti byl lokalizován honeypot. Byl proveden odposlech (na frekvenci 868 MHz), odposlechnutý signál LoRa byl zobrazen v softwaru CubicSDR. Byl zprovozněn vlastní síťový a aplikační server pomocí open-source řešení Chirpstack Server.

Byl vytvořen honeypot využívající vlastní hardware (LoRaWAN gateway postavená na RaspBerry Pi a koncetrátoru ic880a) s možností volby míry interakce (nízká/vysoká). Honeypot nízké míry interakce simuluje službu jen do přihlášení a generuje patřičně logy. Honeypot vysoké míry interakce monitoruje aktivitu útočníka a generuje patřičné logy s využitím HONSSH open-source řešení. Byly popsány a zkoumány dvě varianty high-interaction honeypotu. Jedna bez využití sandboxingu, s využitím firewall a druhá s využitím sandboxingu realizované pomocí chroot/jail. Výsledkem práce je honeypot pro IoT protokol LoRaWAN s volbou mírou interakce: vysoká/nízká.

Literatura

[1] Raza, Kulkarni, Sooriyabandara. *Low power wide area networks: An overview.* IEEE Communications Surveys & Tutorials, 2019, s. 855–873.

[2] Feng Zhang, Shijie Zhou, Zhiguang Qin and Jinde Liu, *Honeypot: a supplemented active defense system for network security* Chengdu:China, 2003, s. 231-235.

[3] *LoRaWAN What is it?: A technical overview of LoRa and LoRaWAN.“ LoRa Alliance, 2015, s. 20 [cit. 11. 10. 2019].* Dostupné z URL: <`https://lora-alliance.org/sites/default/files/2018-04/what-is-lorawan.pdf`>

[4] *Lora-alliance [online]. [cit. 29. 10. 2019].* Dostupné z URL: <`https://lora-alliance.org/`>

[5] *Notes of the IoT provider. Activation and security in LoraWAN [online].[cit. 01. 11. 2019].* Dostupné z URL: <`https://habr.com/ru/post/413105/`>

[6] *LoRaWAN Network: Secure [online]. [cit. 01. 11. 2019].* Dostupné z URL: <`http://www.iksmedia.ru/articles/5573226-Set-LoRaWAN-bezopasnost-obespechiva.html`>

[7] *LoRaWAN: vulnerability analysus and practical by Xueying Yang [online].[cit. 07. 11. 2019].* Dostupné z URL: <`http://repository.tudelft.nl/`>

[8] Mayur Ramgir *Internet of Things.* PEARSON: INDIA, 2019. 392 s. ISBN 10: 9353438942.

[9] Shancang Li and Li Da Xu *Securing the Internet of Things.* Elsevier: Netherlands, 2017. 49-68 s. ISBN 978-0-12-804458-2.

[10] E. Aras, G. S. Ramachandran, P. Lawrence and D. Hughes *Exploring the Security Vulnerabilities of LoRa.* IEEE International Conference on Cybernetics (CYBCONF), Exeter, 2017. 1-6 s. doi: 10.1109/CYBConf.2017.7985777.

[11] Chaojie Gu, Linshan Jiang, Rui Tan, Mo Li, Jun Huang, Nanyang *Attack-Aware Data Timestamping in Low-Power Synchronization.* Technological University, Singapore Peking University, China, 2020. 1-12 s. arXiv:1905.01679.

[12] JOSHI, R. C. a Anjali SARDANA. *Honeypots: a new paradigm to information security.* Boca Raton, FL: Distributed by CRC Press, 2011. ISBN 1578087082.

[13] *Welcome to Honeypots: Monitoring and Forensics [online].[cit. 10. 11. 2019].* Dostupné z URL: <`http://honeypots.sourceforge.net/`>

[14] SPITZNER, Lance. *Honeypots: tracking hackers.* Boston: Addison-Wesley, 2003. ISBN 0321108957.

[15] The Honeynet Project. *Know your enemy: learning about security threats.* Boston: Addison-Wesley, 2004. ISBN 0321166469.

[16] A. Acien, A. Nieto, G. Fernandez, J. Lopez. *A comprehensive methodology for deploying IoT honeypots.* 15th International Conference on Trust, Privacy and Security in Digital Business, 2018 vol. LNCS 11033, pp. 229243, 2018.

[17] *Telnet IoT honeypot. [online]. GitHub: Phype.[cit. 01. 12. 2019].* Dostupné z URL: <https://github.com/Phype/telnet-iot-honeypot/>

[18] *HoneyThing. [online].GitHub: omererdem[cit. 01. 12. 2019].* Dostupné z URL: <https://github.com/omererdem/honeything/>

[19] Y. M. P. Pa, S. Suzuki, K. Yoshioka, T. Matsumoto, T. Kasama, and C. Rossow *Iotpot: analysing the rise of iot compromises.* EMU, vol. 9, p. 1, 2015.

[20] *Dionea. [online].GitHub: DinoTools[cit. 01. 12. 2019].* Dostupné z URL: <https://github.com/DinoTools/dionaea/>

[21] S. Dowling, M. Schukat and H. Melvin. *A ZigBee honeypot to assess IoT cyberattack behaviour.* 2017 28th Irish Signals and Systems Conference (ISSC), Killarney, 2017, pp. 1-6, doi: 10.1109/ISSC.2017.7983603.

[22] *ACTUATOR: SMART PLUG nke WATTECO [online] [cit. 02. 12. 2019].* Dostupné z URL: <http://www.nke-watteco.com/product/actuator-smart-plug/>

[23] *LUMINOSITY SENSOR nke WATTECO [online] [cit. 02. 12. 2019].* Dostupné z URL: <http://www.nke-watteco.com/product/lora-temperature-humidity-sensor/>

[24] *Ic880a-gateway. GitHub [online] [cit. 01. 12. 2019].* Dostupné z URL: <https://github.com/ttn-zh/ic880a-gateway/>

[25] *Chirpstack [online] [cit. 01. 12. 2019].* Dostupné z URL: <https://www.chirpstack.io/>

[26] *HonSSH. GitHub [online] [cit. 01. 05. 2020].* Dostupné z URL: <https://github.com/tnich/honssh/>

Seznam symbolů, veličin a zkratek

ABP	Activation By Personalization
ACK	Acknowledgement
AES	Advanced Encryption Standard
AppKey	Application Key
CRC	Cyclic redundancy check
HSM	Hardware security module
HTTP	HyperText Transfer Protocol
IoT	Interner of Things
IPsec	IP Security
LPWAN	Low Power Wide Area Networks
MIC	Message Integrity Check
MQTT	Message queuing telemetry transport
NVRAM	Nonvolatile random-access memory
NwkKey	Network Key
OTAA	Over The Air Activation
SSH	Secure Shell
SSID	Service Set Identifier
SPI	Serial Peripheral Interface
TCP	Transmission Control Protocol
TLS	Transport layer security

A Obsah přiloženého CD

/ ..... kořenový adresář přiloženého CD
└── `prace.pdf` ..... text bakalářské práce