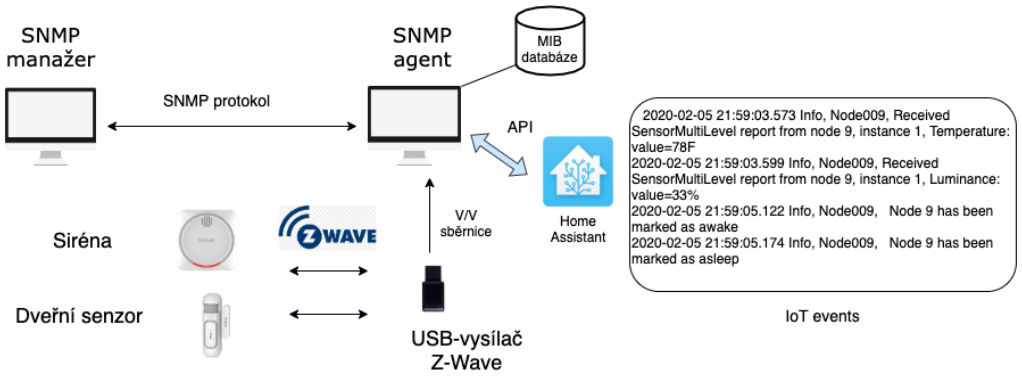
BRNO UNIVERSITY OF TECHNOLOGY
FACULTY OF INFORMATION TECHNOLOGY

Excel@FIT 2020
http://excel.fit.vutbr.cz

# Monitorování IoT sítě pomocí SNMP

Kateryna Polishchuk*




**Abstrakt**
Tato práce řeší implementaci SNMP agenta pro monitorování IoT zařízení, komunikujících přes radiový signál Z-Wave. Podobná zařízení nepodporují protokol SNMP, proto podstatou agenta je shromažďovat informaci o zařízeních z IoT komunikace a ukládat je do proměnných v databázi MIB. Postup implementace agenta bude demonstrován na zařízeních dveřní senzor a signalizační zařízení ze sady inteligentní domácí bezpečnosti D-Link. Zařízení komunikují s USB-vysílačem Z-Wave, který je připojen k počítací, a posílají o sobě události, jako například teplota, stav, typ senzoru atd. Pro vytvoření vlastního agenta SNMP byl použit nástroj Net-SNMP. Data o zařízeních jsem zpracovávala z nástroje Home Assistant, vyhodnocovala jsem události o zařízeních a ukládala informaci do databáze MIB. Vytvořené řešení poskytuje možnost kdykoliv zjistit aktuální stav zařízení pomocí standardních SNMP manažerů. Výsledná práce může pomoct organizacím používajícím IoT zařízení předejít nežádoucímu ohrožení IoT sítě. Výsledkem této práce je rozšíření Z-Wave komunikace o systém SNMP.

**Klíčová slova:** SNMP — IoT zařízení — Monitorování sítě

**Přiložené materiály:** Kód ke stažení

* xpolis03@fit.vutbr.cz, *Faculty of Information Technology, Brno University of Technology*

## 1. Úvod

**[Motivace]** Téma Internet věcí (IoT) se stává každý rok populárnější. IoT zařízení slouží pro automatizaci a bezpečnost domácnosti, zejména pro řízení a správu obytných a komerčních budov. Internet věcí umožňuje zařízením, aby byly vzdáleně kontrolovány pomocí existující infrastruktury. Nicméně v dnešní době vznikají problémy s bezpečnosti IoT zařízení. Možnými hrozbami jsou například útoky na rozbočovač, zapnutí či vypnutí IoT zařízení, zničení zařízení, sledování domácího života. Vzdálená kontrola zařízení pomocí existujících komerčních nástrojů není dostatečná pro monitorování a zjištění skutečného stavu zařízení. Většina IoT zařízení komunikuje s kontrolérem pomocí bezdrátové komunikace typu ZigBee, Bluetooth, Z-Wave apod, což neumožňuje použití tradičních monitorovacích technik jako je System Network Management Protocol (SNMP) [1] či NetFlow [2], které pracují nad TCP/IP. Tento článek představuje návrh implementace SNMP pro IoT sítě.

V současné době ukládání a zpracování dat o IoT zaří-

zeních probíhá ne na samotném zařízení, ale v cloudu. Používání cloudu je důležité pro agregaci a analýzu dat, což při velkém množství dat může výrazným způsobem zlepšit výpočetní výkon. Senzory a zařízení shromažďují data a provádějí akce, samotné zpracování a analýza obvykle probíhá v cloudu. Přístup do cloudu umožňují mobilní aplikace, které poskytují jednotlivé výrobce IoT zařízení. Uchování dat na cloud serverech má svoje nevýhody, například úniky a poškození dat, útoky na servery. K zpracování datových proudů slouží cloudová brána, která filtruje a případně blokuje data před provedením analýzy na cloudu, což způsobuje riziko ztráty dat. Data přenášena v rámci protokolů nemusí být zabezpečená. Útočník může napadnout síť a vytvořit spojení mezi zařízením a bránou nebo mezi bránou a serverem. Další nevýhodou této technologie je nemožnost získat informace o stavu IoT sítě mimo dat z cloudu. Vhodným řešením je proto zabezpečit bezdrátovou komunikaci monitorovacím systémem, který umožní sběr dat ze zařízení lokálně.

Pro správu a monitorování síťových zařízení se používá systém SNMP [1], který umožňuje průběžný sběr nejrůznějších dat pro potřeby správy sítě a jejich následné vyhodnocování. IoT zařízení většinou nepodporují protokol SNMP, neboť nepoužívají TCP/IP ale komunikují přes radiový signál, například Z-Wave. Pro zajištění aktuální informace o stavu IoT zařízení je vhodné rozšířit IoT komunikaci o podporu systému SNMP. Tato podpora může pomoci organizacím používajícím IoT zařízení předejít nežádoucímu ohrožení IoT sítě.

Navržené řešení monitoruje zařízení komunikují přes protokol Z-Wave pomocí agenta SNMP tak, aby bylo možné sledovat z řídící stanice stav IoT zařízení. Postup bude demonstrován na zařízeních siréna a dveřní senzor ze sady inteligentní domácí bezpečnosti D-Link[1], které komunikují s USB-vysílačem přes radiový signál Z-Wave.

# 2. Systém SNMP

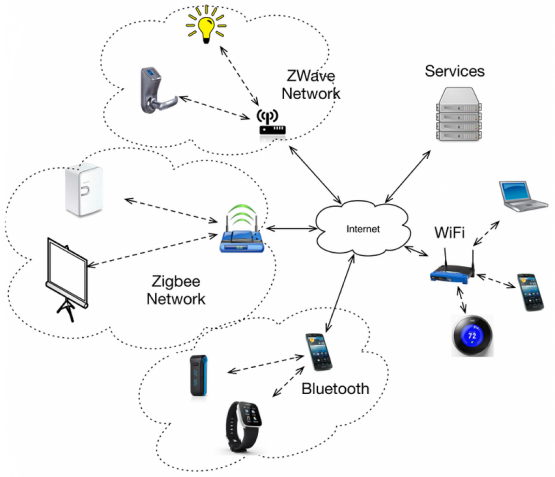
Systém SNMP označuje souhrn specifikací pro správu sítě, které tvoří samotný protokol SNMP, definici datových struktur a související koncepty [3]. Model správy sítě, který se používá pro správu sítě TCP/IP, zahrnuje následující klíčové prvky:

- monitorovací systém – řídící stanice (Network Management System, SNMP server) a SNMP agent,
- monitorované objekty definované pomocí jazyka Structure of Management Information (SMI) [4],
- uspořádání objektů do skupin, decentralizovaná správa objektů – databáze objektů MIB (Management Information Base) a
- komunikační protokol SNMP.

Systém SNMP vytváří strukturu agentů a řídící stanici NMS, která sleduje stav sítě. Řídící stanice a agenti komunikují pomocí protokolu SNMP, který pracuje na aplikační vrstvě. Agenti shromažďují informace o zařízeních a zapisují data do proměnných v databázi Management Information Base (MIB) [5]. Každý objekt je popsán ASN.1 a má své jméno, syntax a kódování. Řídící stanice provádí monitorovací funkci získáním hodnoty objektů MIB. Komunikace mezi entitami protokolu se provádí výměnou zpráv přenášených v rámci jednoho UDP datagramu pomocí základních pravidel kódování ASN.1. Zpráva se skládá z verze identifikátoru, názvu komunity SNMP a datové jednotky protokolu (PDU) [1].

## 2.1 Struktura IoT sítě

Na obrázku č.1 je zobrazená obecná struktura bezdrátové komunikace. Protokol Z-Wave a Zigbee využívají například termostaty, senzory pohybu, alarmy, osvětlení, klimatizace, IP kamery, zámky na dveře, ovladače audio/video techniky atd. Technologie Z-Wave a Zigbee používají smíšenou topologii (mesh), mají nízký výkon a jsou navržené pro přenos malého množství dat na krátké až střední vzdálenosti. Z-Wave využívá síťové pásmo menší než 1 GHz (frekvence se liší v různých zemích), což umožňuje bezkonfliktní provoz při souběhu s Wi-Fi a dalšími systémy založenými na standardu IEEE 802.11. Technologii Bluetooth využívají mobilní telefony, tablety, hodinky, sluchátka, počítače, USB atd. Bluetooth nepodporuje žádný vzdálený přístup, jen point-to-point. Nevýhodou




**Obrázek 1.** Přenos dat mezi prvky systému [6]

[1] Informace o produktu D-Link Security Kit

této technologie je přenos dat na krátkou vzdálenost, maximálně do 10 metrů, protože se k přenosu dat používá velmi malý výkon. Technologie Wi-Fi souvisí s připojením počítačů, tabletů a mobilních telefonů k internetu. Výhodou Wi-Fi technologie je její snadná integrace, možnost připojení i mimo běžné pracovní prostředí v případě propojení s veřejnou sítí a snadné rozšiřování sítě při přidávání dalších klientů.

V současném ekosystému IoT lze různé komponenty IoT široce rozdělit do tří tříd: uzly senzorů, uzly brány a služby IoT. Typické uzly senzorů sestávají z domácích spotřebičů nebo senzorů sledujících fyzické prostředí, které mají nízké výpočetní zdroje, přísná omezení energie a omezené komunikační zdroje. Uzel brány funguje jako agregátor dat senzorů a poskytuje připojení k jiným uzlům senzorů a poskytovatelům služeb. Služby IoT shromažďují data z různých uzlů brány, zpracovávají je v cloudu a poskytují služby specifické pro uživatele nebo události pomocí grafického rozhraní, oznámení nebo aplikace [6]. Chytré domácí mobilní aplikace pomáhají řešit problém správy více zařízení v inteligentních domácnostech z jednoho centrálního uživatelského ovládání. Uživatelé mohou například sledovat stav zařízení a upravovat ho podle svých požadavků vzdáleně. Nevýhodou technologie je nemožnost získat stav zařízení mimo dat z cloudu, protože technologie nepodporuje jiný monitorovací systém.

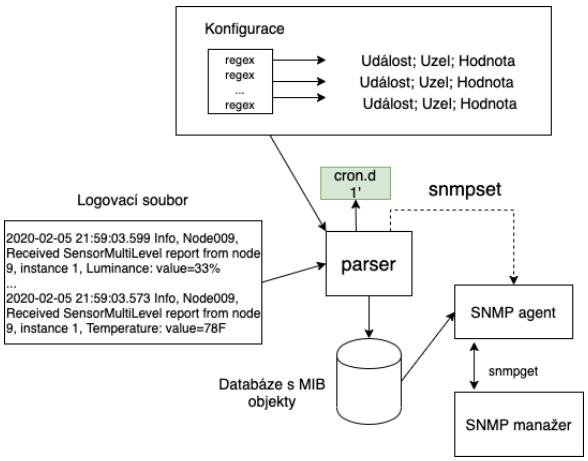
# 3. Návrh monitorování systému

Jeden z požadavků na navrhovaný systém je definice základních prvků, které budou systém tvořit. Pro demonstraci funkčnosti aplikace budeme používat testovací sadu, která obsahuje dveřní senzor a sirénu.

Zařízení posílají události o sobě přes USB-vysílač Z-Wave do nástroje Home Assistant[2]. Jsou to události typu stav zařízení, typ notifikace, typ senzoru, teplota, svítivost. Úkolem agenta je periodicky sbírat data z IoT komunikace, vyhodnocovat události (například pomoci předem vytvořených regulárních výrazu) a ukládat získané nejnovější hodnoty o zařízeních do hodnot proměnných v MIB databázi. Celkově navrhovaný systém se bude skládat z následujících částí:

- Konfigurační soubor – seznám pravidel pro výběr událostí z logu IoT kontroléru.
- Parsovací skript, který prochází logovací soubor, vyhodnocuje události pomocí vytvořených syntaktických pravidel a získané hodnoty ukládá do hodnot proměnných v databázi MIB.
- Démon `cron.d` spouští skript periodicky, například jednou za minutu.
- SNMP agent, který čte hodnoty objektů z MIB databáze a odpovídá na požadavky manažera vracením hodnot těchto objektů.
- MIB databáze s objekty.

Návrh systému je na obrázku č.2:




**Obrázek 2.** Přenos dat mezi prvky systému

Parsovací skript nebo jinak řečeno analyzátor nemusí běžet na stejném stroji s agentem, ale agent mu může poskytnout vzdálený přístup.

# 4. Implementace

Pro implementaci SNMP jsme použili knihovnu Net-SNMP. Net-SNMP[3] je nástroj pro vývoj nových aplikací SNMP v jazyku C, který podporuje rozšíření hlavního agenta za použití dynamicky načtených modulů, externích shell a Perl skriptů a protokolu AgentX. Pomocí Net-SNMP jsme vytvořili agenta, který spravuje IoT objekty ve své MIB databázi.

## 4.1 Sběr dat z IoT komunikace

Data z IoT komunikace ukládá do logovacího souboru nástroj Home Assistant. Každý řádek souboru obsahuje časovou značku události, uzel (zařízení) pro který je vytvořená událost, typ události a samotnou zprávu. Níže je uveden jeden řádek logovacího souboru:

```
2020-02-08 15:25:22.025 Detail, Node011,
Received: 0x01, 0x09, 0x01, 0x41, 0xd3,
0x9c, 0x01, 0x04, 0x10, 0x05, 0xe9
```

Některé aplikační zprávy mají formát 16-bitových hodnot a většinou je struktura následující [7]:

- 0x01 (1B, začátek zprávy)
- délka zprávy (1B)
- typ zprávy (1B, 0x00 - request, 0x01 - response)

[2] Instalace nástroje https://www.home-assistant.io/getting-started/

[3] Net-SNMP http://www.net-snmp.org/wiki/index.php/Tutorials#Coding_Tutorials

  - funkce (1B, 0x04, 0x13)
  - ID úzlu (1B)
  - délka dat (1B)
  - typ příkazu (1B, např. 0x84 Wakeup atd.)

Každý uzel také obsahuje seznam příkazů, které reprezentují atributy každého zařízení, například:

- COMMAND_CLASS_BASIC
- COMMAND_CLASS_SWITCH_BINARY
- COMMAND_CLASS_SENSOR_BINARY atd.

Nevýhodou nástroje Home Assistant je nečitelnost a složitá interpretace příkazu z logovacího souboru. Proto je potřeba implementovat formální pravidla, pomocí kterých se události vyhodnotí a uloží do MIB databáze v čitelném formátu. Úkolem této fáze je vybrat vhodné objekty, které chceme u zařízení monitorovat, definovat formát příkazů a vytvořit konfigurační soubor, který bude obsahovat regulární výrazy (dále jen RV) pro získání hodnot objektů. Soubor je možné sestavit ve formátu `csv` nebo `json`, aby ho bylo možné jednoduše parsovat.

Vytvořené řešení obsahuje konfigurační soubor, ve kterém každý řádek obsahuje RV události, název MIB objektu a RV příkazu, oddělené středníkem ve formátu `csv`.

Konfigurační soubor je pak vstupem pro parsovací skript, jinak řečeno analyzátor, který prochází nejnovější události logovacího souboru a hledá pro ně odpovídající pravidlo pro interpretaci. Analyzátor je navržen a implementován v jazyku Python 3. Pokud se pravidlo našlo, hledá se hodnota objektu z nalezeného řádku a ukládá se do odpovídajícího objektu v databázi MIB. Poslední čas události se ukládá do sdílené paměti, čímž se zajišťuje nalezení aktuální hodnoty události.

Aby aplikace vracela vždycky aktuální hodnoty objektů, je potřeba použít unixovou utilitu pro periodické automatické spouštění skriptu `cron.d`.

Příklad navrženého pravidla vypadá následně:

```
Received SensorMultiLevel report from node
\d{1,2},instance \d{1,2},
Luminance: value=\d{1,3}\%;
sensorLuminance;value=(.+?)\%
```

Pravidlo je určeno pro získaní hodnoty svítivosti u dveřního senzoru. Řádek obsahuje tři části oddělené středníkem, kde první část je RV, který odpovídá události:

```
2020-02-05 21:59:03.599 Info, Node009,
Received SensorMultiLevel report from node
9, instance 1, Luminance: value=33%
```

Druhá část je název MIB objektu *sensorLuminance*, do které se uloží hodnota získaná pomocí regulárního výrazu:

```
value=(.+?)\%
```

Další pravidlo, které získává stav dveřního senzoru, vypadá následně:

```
Node \d{1,2} has been marked as
\b(awake|asleep)\b;sensorState;\bas (.+?)$
```

Pravidlo pro získání stavu sirény:

```
Received SensorBinary report:
Sensor:\d{1,3} State=\b(On|Off)\b;
sirenState;State=(.+?)$
```

Pravidlo pro získání teploty dveřního senzoru:

```
Received SensorMultiLevel report from node
\d{1,2}, instance \d{1,2}, Temperature:
value=\d{1,3}F;sensorTemperature;
value=(.+?)F
```

## 4.2 Implementace SNMP agenta

Nástroj Net-SNMP umožňuje nakonfigurovat vlastní objekty do hlavního SNMP agenta nebo implementovat rozšířeného agenta (subagenta AgentX) s podporou vlastních objektů. Navržené řešení používá subagenta. Je to vlastní proces, který se připojuje k hlavnímu agentovi pomocí protokolu AgentX. Hlavního agenta je potřeba také nakonfigurovat vhodným způsobem, tzn. definovat community string, typ přístupu (read-write, read-only), název hosta, kterému se přístup povoluje, část stromu, ke které lze přistupovat, případně port (implicitně 161), uživatelé (pokud se používá SNMP verzi 3) a IP adresy, které mohou na portu naslouchat.

Rozšířený agent obsahuje implementaci inicializačních modulů pro objekty MIB, tj. přiřazení OID a následující registraci v MIB databázi při spouštění. Agent má hlavní funkci, která odpovídá na požadavky manažera získáním hodnoty MIB objektů.

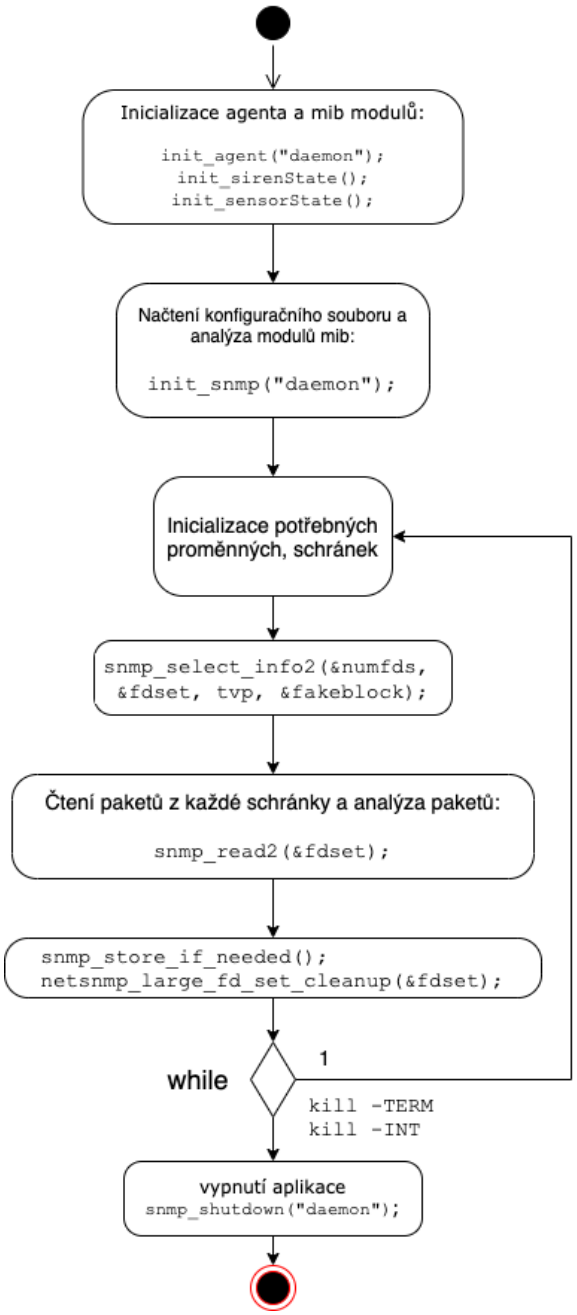
Agent používá knihovny *net-snmp-config, net-snmp-includes, net-snmp-agent-includes, large_fd_set* a *snmp_assert*.

Hlavní funkce kontroluje pakety přicházející na port SNMP a zpracovává je (funkce `snmp_read`), pokud jsou některé nalezeny.

Funkce `snmp_read()` přistupuje k množině schránek a čte z nich pakety, dále je analyzuje. Výsledná datová jednotka protokolu (PDU) je předána do rutiny zpětného volání pro relaci (funkce `snmp_session()`). Pokud se zpětné volání úspěšně vrátí, PDU a její požadavek budou odstraněny. Funkční diagram činnosti agenta je na obrázku č.3:

## 4.3 Objekty SNMP pro monitorování IoT sítě

Nejdůležitější částí pro vytvoření vlastní MIB databáze je seznam objektů, které budou uloženy v MIB. Tyto objekty musí striktně dodržovat syntax ASN.1. Pokud




**Obrázek 3.** Diagram činnosti

chceme definovat vlastní objekty, které chceme sdílet se světem, potřebujeme je zařadit do MIB stromu tak, aby nebyly v konfliktu s jinými objekty. Pokud definujeme objekty, které budeme používat pouze interně, můžeme je vložit do experimentálního stromu. Jen si musíme dát pozor na překrývání OID[4]. Při napsání vlastní MIB jsem použila ukázkový příklad Net-SNMP a definovala vlastní MIB do větve *netSnmpExamples*. Takto vypadá definice jednoho z objektů MIB databáze, konkrétně objekt popisující sirénu:

```
sirenState OBJECT-TYPE
SYNTAX  DisplayString(SIZE(0..3))
MAX-ACCESS  read-write
STATUS      current
DESCRIPTION "Status of the device
may take on values off/on"
DEFVAL { "off" }
::= { iotAgentModules 2 }
```

Tento objekt obsahuje hodnotu typu string o velikosti 0-255 znaků s implicitní hodnotou ”off”. Další objekty, které se použily pro testování funkčnosti agenta, jsou definované následně:
Objekt pro sledování svítivosti:

```
sensorLuminance OBJECT-TYPE
SYNTAX  Integer32
MAX-ACCESS  read-write
STATUS      current
DESCRIPTION
    "Door/Window luminance"
DEFVAL { 0 }
::= { iotAgentModules 4 }
```

Objekt pro sledování teploty:

```
sensorTemperature OBJECT-TYPE
SYNTAX  Integer32
MAX-ACCESS  read-write
STATUS      current
DESCRIPTION
    "Door/Window temperature"
DEFVAL { 0 }
::= { iotAgentModules 5 }
```

# 5. Testování

V práci byl vytvořen systém SNMP, který poskytuje monitorovací funkci zařízením komunikujícím přes radiový signál Z-Wave. Ve virtuálním prostředí běžel agent a rozšířený agent. Zároveň na počítací mimo virtuální prostředí běžel nástroj Home Assistant. K počítači byl také připojený USB-vysílač přes V/v sběrnici, který naslouchal na Z-Wave síti. Jednou za minutu se spouštěl parsovací skript. Při experimentech se SNMP manažer dotazoval agenta na hodnoty objektů zařízení. Bylo vidět, jak se mění hodnoty objektů zařízení, například teplota a svítivost, stav (asleep nebo awake) u dveřního senzoru a stav (On/Off) u sirény.

## Test 1

Logovací soubor obsahuje událost „stav dveřního senzoru“:

```
2020-02-05 21:59:05.122 Info, Node009,
Node 9 has been marked as awake
```

[4] Viz http://net-snmp.sourceforge.net/wiki/index.php/Writing_your_own_MIBs

kterou SNMP agent zpracoval jako:

```
$ snmpget -v2c -c public
192.168.251.2 1.3.6.1.4.1.8072.2.4.1.1.3.0

NET-SNMP-EXAMPLES-MIB::
netSnmpExamples.4.1.1.3.0 = STRING: "awake"
```

### Test 2

Později došlo ke změně hodnoty události:

```
2020-02-05 21:59:05.174 Info, Node009,
Node 9 has been marked as asleep
```

která se hned uložila do databáze

```
NET-SNMP-EXAMPLES-MIB::
netSnmpExamples.4.1.1.3.0 = STRING:
"asleep"
```

### Test 3

Logovací soubor obsahuje událost „teplota senzoru“:

```
2020-02-05 21:59:03.573 Info,
Node009, Received SensorMultiLevel report
from node 9, instance 1,
Temperature: value=78F
```

agent pak uložil hodnotu následně:

```
NET-SNMP-EXAMPLES-MIB::
netSnmpExamples.4.1.1.5.0 = INTEGER: 78
```

### Test 4

Logovací soubor obsahuje událost „svítivost senzoru“, která vypadá následně:

```
2020-02-05 21:59:03.599 Info, Node009,
Received SensorMultiLevel report from node
9, instance 1, Luminance: value=33%
```

do databáze se pak uložila hodnota:

```
NET-SNMP-EXAMPLES-MIB::
netSnmpExamples.4.1.1.4.0 = INTEGER: 33
```

### Test 5

Ukázka události „stav sirény“

```
2020-02-11 14:34:23.336 Info,
Node012, Received SensorBinary report:
Sensor:1 State=Off
```

kterou SNMP agent zpracoval jako:

```
NET-SNMP-EXAMPLES-MIB::
netSnmpExamples.4.1.1.2.0 = STRING: "Off"
```

### Test 6

Dále došlo ke změně hodnoty události:

```
2020-02-11 14:34:23.345 Info, Node012,
Received SensorBinary report: Sensor:1
State=On
```

která se následně zpracovala jako:

```
NET-SNMP-EXAMPLES-MIB::
netSnmpExamples.4.1.1.2.0 = STRING: "On"
```

## 6. Závěr

IoT zařízení v dnešní době používají spousta organizaci pro automatizaci a bezpečnost. Spolu s tím narůstá počet útočníku a hrozeb na zařízení, které neposkytují dostatečnou bezpečnost. Systém SNMP umožňuje správu sítě a průběžný sběr nejrůznějších dat o zařízeních. Je to standard sledovaní řízení bezpečnosti sítě, zpracování chyb a odhalení možných útoků. Některá IoT zařízení tento systém však nepodporují. Přínosem této práce je rozšíření Z-Wave komunikace o systém SNMP. Tento nástroj je důležitý pro organizace používající IoT zařízení pro zkoumání stavu zařízení na útoky, aby je bylo možné včas odhalit. Tato práce popisuje postup vytvoření SNMP agenta v jazyku C, MIB databáze a navrhuje analyzátor, který interpretuje data z logovacího souboru na hodnoty proměnných v databázi MIB. Daný postup lze aplikovat i na jiné IoT sítě, které lokálně používají radiové spojení, například typu Bluetooth, Zigbee apod. Je potřeba pouze adaptér pro danou síť, softwarový nástroj Home Assistant pro zpracování události. Dále je potřeba popsat události, které se budou odchytávat pomocí regulárních výrazů a vytvořit definici objektů MIB.

## 7. Poděkování

Ráda bych poděkovala svému vedoucímu Ing. Petrovi Matouškovi Ph.D., M.A., za odborné rady a vedení při vypracování této práce.

## Literatura

[1] J. Case et al. A Simple Network Management Protocol (SNMP). IETF RFC 1157, 1990.

[2] B. Claise. Cisco Systems NetFlow Services Export Version 9. IETF RFC 3954, 2004.

[3] William Stallings. *SNMP, SNMPv2, SNMPv3, and RMON 1 and 2*. Addison-Wesley, 3. edition, 1998. ISBN: 0-201-48534-6.

[4] M. Rose and K. McCloghrie. Structure and Identification of Management Information for TCP/IP-based Internets. IETF RFC 1155, 1990.

[5] R. Presuhn et al. Management information base (mib) for the simple network management protocol (snmp). IETF RFC 3418, 2002.

[6] Amit Sheth. Semantic gateway as a service architecture for iot interoperability, October 2014. https://arxiv.org/abs/1410.4977.

[7] Onur Dundar. *Home Automation with Intel Galileo*. Packt Publishing Ltd, 3. edition, 2015. ISBN: 1-785-28726-5.