Manuál k prokejtu do ISA

Exportér SNMP Gauge metrik do OpenTelemetry (OTEL)

Adam Žluva (xzluva01)

17. listopadu 2025

Obsah

1 Úvod

Tento projekt si klade za cíl implementovat program, napsaný v jazyce **C++**, `snmp2otel`. Program periodicky zjišťuje stav zařízení na síti pomocí protokolu **SNMP** a exportuje nasbíraná data na **OpenTelemetry** endpoint.

Implementace dále zahrnuje nástroje pro zakódování/dekódování imformací podle formátu **BER** pro **ASN.1**.

2 Návrh projektu

Implementace programu využívá paradigmatu OOP. Jednotlivé části projektu jsou členěny do tříd (modulů), kde každá třídá (modul) plní jeden jednoduchý účel.

Hlavní dotazovací smyčka je tedy jednoduše poskládána z těchto modulárních tříd, čímž vzniká velmi přehledná struktura projektu.

Hlavní dotazovací smyčka programu funguje následovně:

1. Zakódování SNMP Get požadavku pomocí BER
2. Odeslání zakódovaného SNMP Get požadavku přes UDP na zadaného agenta
3. Dekódování odpovědi SNMP agenta
4. Sestavení OTLP/HTTP zprávy
5. Odeslání OTLP/HTTP zprávy přes TCP na zadaný OTEL endpoint

3 Implementace

3.1 Parsování CLI argumentů

Po spuštění se program pokusí přeparsovat zadané argumenty, zkontroluje jejich správnost a sestaví datovou strukturu, která uchovává již pouze validní hodnoty všech parametrů. Tato struktura je definována v `include/arguments.hpp`.

Jestliže uživatel zadá **nevalidní volitelný argument**, program vypíše varování na standardní chybový výstup a dále použije implicitní hodnotu daného argumentu. Vykonávání programu se tedy **nepřeruší**.

Jestliže uživatel **vynechá nebo zadá nevalidní hodnotu povinného argumentu**, program vypíše chybu na standardní chybový výstup a **vykonávání programu je ukončeno**.

Instance `Arguments` je uchovávána v singleton instanci třídy `Context`, pro jednoduchý globální přístup. Třída `Context` původně vznikla s předpokladem potřeby globálního přístupu k vícero strukturám, avšak tohoto nakonec nebylo potřeba.

3.2 Čtení souboru s OID

Po úspěšném parsování CLI argumentů se program pokusí přečíst soubor se seznamem dotazovaných OID. Ke čtení a filtraci OID slouží třída `OIDReader`; definována v `include/oidreader.hpp`.

`OIDReader` otevře soubor a postupně čte jednotlivé řádky. Prázdné řádky a řádky začínající znakem `#` jsou ignorovány. **Mezery jsou z řádku odstraněny**, což může způsobit, že OID zadané následovně: `1.3.6 .1.2. 1. 1.3.0`, bude přečteno jako validní OID `1.3.6.1.2.1.1.3.0`.

Jestliže dojde k chybě při čtení souboru nebo nejsou načteny žádné validní OID, **vykonávání programu je ukončeno**.

3.3 Hlavní smyčka

Hlavní smyčka je implementována přímo v hlavním zdrojovém souboru `src/main.cpp`. Opakuje se, dokud uživatel programu napošle signál **SIGTERM**, **SIGINT**, nebo selže či vyprší komunikace vícekrát, než je zadáno v argumentu *-r* (retries).

3.3.1 Sestavení a odeslání SNMP Get požadavku

Pomocná třída `SNMPHelper` (`include/snmphelper.hpp`) zakóduje SNMP Get požadavek pomocí třídy `Encoder` (`include/encoder.hpp`).

`Encoder` poskytuje metody pro zakódování různých datových struktur do formátu BER, které `SNMPHelper` využívá ke koreknímu sestavení SNMP Get požadavku, definován podle ASN.1.

`SNMPHelper` vrací vektor bajtů, které je následně možné odeslat přes UDP pomocí `UDPClient` (`include/udpclient.hpp`)

3.3.2 Příjem SNMP odpovědi

Při úspěšném přijetí SNMP odpovědi se program pokusí odpověď dekódovat pomocí *SNMPHelper*. Pro dekódování `SNMPHelper` využívá třídu `Decoder` (`include/decoder.hpp`), která podobně jako `Encoder` poskytuje metody pro dekódování bajtů ve formátu BER.

Dekódovanými daty je naplněna pomocná struktura `SNMPResponse` (`include/snmphelper.hpp`).

3.3.3 Export dat na OTEL endpoint

`SNMPResponse` se předá třídě `OTELExporter` (`include/otelexporter.hpp`. `OTELExporter` převede `SNMPResponse` na JSON, pokusí se navázat spojení se zadaným endpointem přes TCP, pomocí třídy `TCPClient` (`include/tcpclient.hpp`), a odešle data.

Jestliže se data nepodaří zapsat, z jakýchkoliv důvodů, počítá se to jako neúspěch iterace. Tedy jestliže se nepodaří data exportovat vícekrát, než je zadáno v argumentu *-r* (retries), program se ukončí.

4 Návod na použití

`snmp2otel -t target [-C community] -o oids_file -e endpoint [-i interval] [-r retries] [-T timeout] [-p port] [-v]`

Program podporuje následující argumenty:

- `-t target` – povinný argument, specifikující adresu dotazovaného SNMP agenta.
- `-C community` – volitelný argument, specifikující textový přístupový řetězec. Implicitně hodnota `"public"`.
- `-o oids_file` – povinný argument, specifikující cestu k soubru se seznamem OID.
- `-e endpoint` – povinný argument, specifikující URL OTEL endpointu.
- `-i interval` – volitelný argument, specifikující čas v sekundách mezi SNMP požadvky. Implicitně hodnota `10`.
- `-r retries` – volitelný argument, specifikující maximální počet opakování požadavku/exportu při neúspěchu. Implicitně hodnota `2`.
- `-T timeout` – volitelný argument, specifikující čas v milisekundách vypršení UDP/TCP komunikace. Implicitně hodnota `1000`.
- `-p port` – volitelný argument, specifikující port pro komunikaci s SNMP agentem. Implicitně hodnota `161`.
- `-v` – volitelný argument, přepínající výpis ladících informací na standardní výstup.

5 Testování

Z časových důvodů nebyly implementovány automatizované testy. Program byl testován manuálně během vývoje (”whitebox”testy, ale v manuální podobě).

Program je schopen zvládnout základní chybové stavy a chová se korektně, pokud je dodržen správný formát všech dat.

Pro manuální testy byl zprovozněn lokální SNMP server pomocí nástroje **snmpd** a lokální OTEL enpoint uvnitř **docker** kontejneru.

5.1 Demonstrace

Obrázek 1: Spuštění programu

Program byl spuštěn s konfigurací, dotazující pouze jedno OID (sysUpTime). Je možné vidět, že SNMP agent odpověděl s touto hodnotou:

```
Decoded SNMP Response:
requestId=0 errorStatus=0 errorIndex=0
1.3.6.1.2.1.1.3.0 = 12195
```

Kontejner s běžicím OTEL exportovaná data úspěšně získal, jak můžeme vidět na Obrázku 2.

```
Metric #0
Descriptor:
     -> Name: snmp_1_3_6_1_2_1_1_3_0
     -> Description: SNMP metric for OID 1.3.6.1.2.1.1.3.0
     -> Unit: nope
     -> DataType: Gauge
NumberDataPoints #0
Data point attributes:
     -> oid: Str(1.3.6.1.2.1.1.3.0)
StartTimestamp: 1970-01-01 00:00:00 +0000 UTC
Timestamp: 2025-11-17 22:06:11.233202931 +0000 UTC
Value: 12195.000000
```

Obrázek 2: Výstup OTEL

6 Reference

Z časových důvodů této části nebylo věnováno mnoho času...

- RFC 3416–3418
- ITU-T X.690
- OpenTelemetry Specification — Metrics, OTLP/HTTP
- GitHub Rpozitář OpenTelemetry
- Wikipedia: Basic Encoding Rules