Radosław GŁĄB[1], Dorota Rabczuk[2]

Uniwersytet Morski w Gdyni, Wydział Elektryczny (1), Uniwersytet Morski w Gdyni, Zakład Telekomunikacji Morskiej (2)

doi:10.15199/48.2022.09.32

Aplikacja do monitorowania stacji bazowej systemu radiokomunikacyjnego przy użyciu protokołu SNMP

Streszczenie. *W artykule zaprezentowano autorskie rozwiązanie aplikacji do monitorowania stacji bazowej systemu radiokomunikacyjnego. Aplikacja monitoruje i archiwizuje parametry stacji, a w przypadku wykrycia stanów alarmowych wysyła powiadomienia za pomocą e-mail lub komunikacji MQTT (Message Queueing Telemetry Transport) do urządzeń w systemie Android. Aplikacja działa na zdalnym komputerze i jest połączona tunelem VPN (Virtual Private Network) z agentem SNMP (Simple Network Management Protocol) stacji bazowej. Aplikacja została przetestowana ze stacją bazową systemów radiokomunikacyjnych Tetra i DMR.*

Abstract*. The article presents an application which monitors the base station of a radio communication system. The application monitors and archives the station's parameters and sends notifications via email or MQTT (Message Queueing Telemetry Transport) to Android devices when alarm states are detected. The application runs on a remote computer and is connected with the SNMP (Simple Network Management Protocol) agent of the base station via a secure VPN (Virtual Private Network) tunnel. The application was tested with the base station of the Tetra and DMR radio systems. (**Application to monitor the base station of a radio communication system using the SNMP protocol**)*

Słowa kluczowe: protokół SNMP, net-snmp, system NMS
Keywords: Simple Network Management Protocol, net-snmp, Network Management System

Wstęp

Zapewnienie stabilnych połączeń w sieci Internet wymaga ciągłego automatycznego monitorowania wszystkich urządzeń sieciowych. W przypadku złożonej infrastruktury sieciowej dane z monitoringu powinny zostać przesyłane przez sieć do jednego stanowiska – aplikacji zarządzającej. Prostym i skutecznym standardem monitorowania i zarządzania siecią TCP/IP jest protokół SNMP (Simple Network Management Protocol) [1].

Na kluczowe komponenty systemu SNMP składają się: sieciowe urządzenie monitorowane z rezydującym na nim agentem SNMP, baza informacji MIB (Management Information Base) oraz aplikacja zarządzająca uruchamiana na urządzeniu sieciowym (komputer PC, smartfon, router).

Agent SNMP jest komponentem softwarowym działającym na zarządzanym urządzeniu, który utrzymuje komunikację ze zdalnym hostem. Do jego zadań należy: pobieranie wartości bieżących monitorowanych obiektów i zdarzeń, odpowiadanie na żądania hosta i przesyłanie danych, operacje w bazie danych MIB, wysyłanie powiadomień (TRAP) w przypadku wystąpienia nagłego zdarzenia lub błędu na monitorowanym urządzeniu.

Baza informacji MIB zawiera zmienne monitorowanego urządzenia. MIB wykorzystuje strukturę drzewiastą do przechowywania danych. Węzeł drzewa wskazuje obiekt monitorowany, który jest identyfikowany przez ścieżkę rozpoczynającą się od katalogu głównego nazywaną identyfikatorem obiektu OID (Object Identifier).

Aplikacja zarządzająca nadzoruje pracę agentów SNMP na monitorowanych urządzeniach, wysyła żądania, odbiera pułapki (TRAP), wykrywa nowe urządzenia w sieci, analizuje wydajność urządzeń.

W protokole SNMP zdefiniowano pięć wiadomości, które mogą być przesyłane między agentem i hostem: żądanie GET do pobrania od agenta bieżącej wartości zarządzanego obiektu, żądanie GETNEXT do pobrania bieżącej wartości kolejnego obiektu, żądanie SET do uaktualnienia wartości obiektu, odpowiedź na żądania GET, GETNEXT, SET wysyłana zwrotne przez agenta, wiadomość TRAP zawierająca informację alarmową o stanie obiektu wysyłana samorzutnie przez agenta [2, 3].

Drzewo MIB bazy informacji stacji bazowej Tetra i DMR

Prezentowana aplikacja monitoruje stację bazową systemu radiokomunikacyjnego Tetra (Terrestrial Trunked Radio) i DMR (Digital Mobile Radio).

Na kontrolerze BSC stacji bazowej (Base Station Controller) zainstalowano oprogramowanie agenta SNMP udostępnione przez producenta. Załączona dokumentacja producenta zawiera strukturę drzewa MIB, w którego gałęziach rozmieszczone są obiekty OID (Object ID) przechowujące dane konfiguracyjne i stany pracy urządzenia. Obiekty te są identyfikowane ciągiem cyfr reprezentujących trasę od pnia do wybranej gałęzi drzewa MIB. Od pnia drzewa MIB kontrolera stacji bazowej rozchodzą się trzy główne konary. W pierwszym są przechowywane dane rejestracyjne stacji, drugi grupuje obiekty reprezentujące daną funkcjonalność np. rejestratora rozmów. Trzeci konar rozdziela się na gałęzie, w zakończeniach których umieszczono pojedyncze obiekty reprezentujące parametry stacji. Układ obiektów w drzewiastej strukturze MIB powinien optymalizować czas ich odczytu i ułatwić dotarcie żądaniem GETNEXT do obiektów reprezentujących pokrewną funkcjonalność na zasadzie przyporządkowania im kolejnych numerów OID (Object Identifictors) np. 1.3.6.1.4.1.15228.3.1.2.1.1.1, 1.3.6.1.4.1.15228.3.1.2.1.1.2,, 1.3.6.1.4.1.15228.3.1.2.1.1.3 itd. Kolejne numery OID przypisano np. obiektom zawierającym: numer i nazwę kontrolera, wersję hardwarową i softwarową stacji, datę aktywacji [4,5].

Dla tworzonej aplikacji najistotniejsze są obiekty reprezentujące stany pracy systemu radiokomunikacyjnego w tym: transceivera radiowego, bramy głosowej, bramy pakietowej transmisji danych, bramy aplikacji, bramy terminalowej. Obiekty te są indeksowane numerem węzła sieci oraz pracującej w tym węźle stacji BSC. Każdy obiekt może znajdować się w jednym z czterech stanów: OK, Ostrzeżenie, Alarm, Blokada lub Nie skonfigurowany.

Narzędzia softwarowe wykorzystane przy tworzeniu aplikacji monitorującej

Założono wykonanie aplikacji internetowej działającej na serwerze, dla której nie jest wymagana instalacja na zdalnym komputerze, a do dostępu wystarczy jedynie połączenie z Internetem.

Projekt zrealizowano w środowisku wirtualnym VirtualBox na linuxowym systemie operacyjnym gościa maszyny fizycznej. Zastosowanie wirtualizacji w prezentowanej aplikacji zarządzającej zapewnia elastyczność rozwiązania, aplikacja działająca na serwerze

wirtualnym może zostać zaimplementowana w dowolnym środowisku sprzętowym pod nadzorem VirtualBoxa. Spakowany obraz maszyny wirtualnej można udostępnić kolejnym klientom, dzięki czemu nie trzeba konfigurować każdego stanowiska zarządzania oddzielnie. Rozwiązanie zapewnia też bezpieczeństwo eksploatacji - każda maszyna wirtualna może zostać w dowolnym momencie zamrożona, uruchomiona, skopiowana oraz backupowana. W razie potrzebnych modernizacji obraz systemu można skopiować, a prace modernizacyjne przeprowadzić na kopii bez ingerencji w środowisko produkcyjne.

Aplikacja zarządzająca została napisana w multi-edytorze VSC (Visual Studio Code), z którego można korzystać w systemach operacyjnych Linux, Windows i macOS. VSC wspiera 30 języków programowania w tym zastosowane podczas pisania Python, HTML, CSS i JavaScript. Strona internetowa powstała w oparciu o framework Django dostarczający narzędzi do tworzenia dynamicznych stron internetowych. Językiem programowania silnika aplikacji jest Python – język interpretowany, ze swej natury przeznaczony do tworzenia aplikacji interaktywnych.

Podczas pisania kodu programu wykorzystano pakiet Net-SNMP wdrażający protokół SNMP w wersjach v1, v2c i v3. Do pobrania informacji z urządzenia przy użyciu pojedynczych żądań służą w nim polecenia snmpget oraz snmpgetnext, a w przypadku wielu żądań polecenie snmpwalk. Do konfiguracji urządzenia dostępna jest komenda snmpset. Pakiet NET-SNMP oferuje możliwość konwersji między numeryczną a tekstową formą identyfikatorów obiektów OID. Do wyświetlania zawartości i struktury bazy MIB przeznaczone jest polecenie snmptranslate.

Implementacje poleceń pakietu Net-SNMP można znaleźć np. w bibliotece pysnmp, którą doinstalowano na potrzeby projektu [6].

Baza danych MIB

Odebrane metryki SNMP są przechowywane w obiektowo-relacyjnej bazie danych. Wybrano bazę PostgreSQL która działa pod systemami operacyjnymi Linux, Microsoft Windows oraz Apple MacOS wpisując się w założenie o możliwości przeniesienia aplikacji na dowolną platformę sprzętową.

Na potrzeby projektu w bazie PostgreSQL zostało utworzonych 5 tabel, które przechowują:

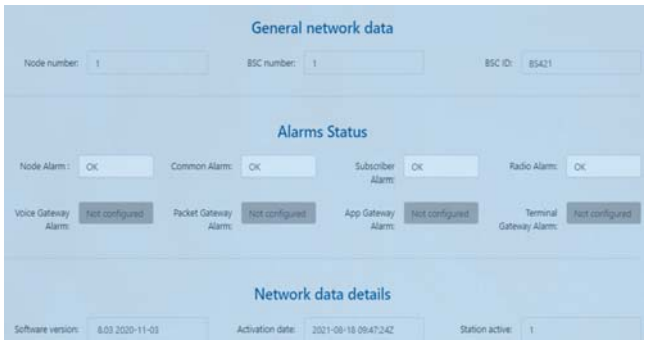
- podział metryk stacji bazowej na grupy, w których są analizowane,
- stany monitorowanych obiektów OID,
- archiwum stanów alarmowych,
- dane wyświetlane w oknie głównym aplikacji,
- adres IP stacji monitorowanej, adres email na który są wysyłane powiadomienia oraz częstotliwość wysyłania powiadomień w sieci MQTT.

Funkcjonalność autorskiej aplikacji monitorującej

Funkcjonalność aplikacji odpowiada potrzebom zdalnego monitorowania stacji bazowej systemu Tetra i DMR w Laboratorium Systemów Radiokomunikacji Ruchomej UMG w Gdyni.

Do podstawowych funkcji zaliczono: programowanie adresu IP stacji monitorowanej, wybór monitorowanych obiektów i wyświetlanych alarmów, cykliczny automatyczny odczyt stanów obiektów oraz alarmów z różnych gałęzi drzewa MIB, zapis logów ze stanami alarmowymi i ostrzegawczymi w bazie danych wraz ze znacznikami czasu, przechowywanie logów w bazie danych przez okres pięciu dni i automatyczne kasowanie starszych wpisów w bazie, wybór alarmów wymagających powiadomień e-mailowych, wysyłanie powiadomień alarmowych i ostrzegawczych na wybrany adres e-mail.

Do dodatkowych opcjonalnych funkcjonalności zaliczono możliwość przesyłania stanów alarmowych za pośrednictwem sieci MQTT dającą użytkownikowi możliwość sprawdzania na bieżąco stanów stacji bazowej w aplikacji pod systemem Android, która po skonfigurowaniu utrzymuje połączenie z systemem monitoringu.

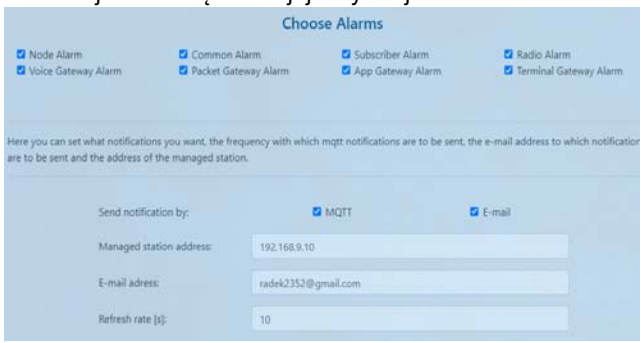


Rys.1. Strona główna aplikacji zarządzającej

Interfejs autorskiej aplikacji monitoringu składa się z trzech zakładek: *System parameters, Configuration oraz Logs* widocznych na górnym pasku okna na rysunku 1, przy czym *System parameters* pełni rolę strony głównej.

Informacje wyświetlane na stronie głównej interfejsu (rys.1) zostały podzielone na trzy podtematy:

- *General network data* zawiera informacje o numerze monitorowanego węzła, numerze i identyfikatorze stacji bazowej systemu radiokomunikacyjnego,
- *Alarms data* pokazuje aktualne stany wszystkich obiektów reprezentujących alarmy stacji bazowej,
- *Network data details* wyświetla numer wersji firmwaru stacji oraz datę i czas jej aktywacji.



Rys.2. Widok zakładki Configuration z wybraną opcją Notification

Zakładka *Configuration* została podzielona na dwie podstrony, pierwsza z nich *Display* umożliwia wybór danych wyświetlanych w oknie głównym aplikacji.

Druga podstrona *Notification* umożliwia skonfigurowanie powiadomień e-mailowych oraz wysyłanych w sieci MQTT (rys.2), w szczególności:

- wybór rodzaju powiadomień: e-mailowe, MQTT lub oba,
- zaprogramowanie adresu IP monitorowanej stacji bazowej,
- wskazanie adresu e-mail do wysyłania powiadomienia
- wybór częstotliwości odświeżania powiadomień MQTT (*Refresh rate)*

Zakładka *Logs* umożliwia użytkownikowi wybór zakresu data/czas do prezentacji logów, wybór jest dokonywany na komponencie kalendarza/zegara.

Powiadomienia e-mailem

Dla wysłania powiadomień o stanach alarmowych e-mailem został napisany skrypt realizujący następujące funkcje:

- pobranie stanów obiektów OID
- wybór obiektów, których stany alarmowe wymagają powiadomień,
- sprawdzenie czy aktualny stan obiektu różni się od ostatniego zapisanego w bazie danych
- sprawdzenie czy aktualny stan obiektu jest jednym ze stanów ostrzegawczych lub alarmowych (Alarm, Warning, Blocked).
- utworzenie szablonu wiadomości e-mail, zdefiniowanie adresu e-mail oraz danych serwera pocztowego.
- wysłanie powiadomienia e-mail i zapisanie stanów alarmowych w bazie danych pod postacią logu.

Wysyłanie wiadomości e-mail zrealizowano w oparciu o biblioteki smtplib i ssl, implementujące protokoły sieciowe SMTP (Simple Mail Transfer Protocol) oraz SSL (Secure Sockets Layer). Moduł interfejsu poczty e-mail utworzono z wykorzystaniem bibliotek MIMEText oraz MIMEMultipart,

Cykliczne wysyłanie powiadomień e-mail szybko zapełnia skrzynkę odbiorczą użytkownika, dlatego zdecydowano o jednokrotnym wysyłaniu powiadomień tylko w przypadku zmiany stanu na *Warning, Alarm* lub *Blocked* lub zmiany powrotnej do stanu *OK*.

Aplikacja została przetestowana dla wybranych stanów ostrzegawczych i alarmowych stacji BSC systemu radiokomunikacyjnego Tetra/DMR.

W oknie aplikacji stan alarmowy obiektu jest sygnalizowany jaskrawym kolorem (rys.3). Stan obiektu jest wpisywany do logów w bazie danych, a równocześnie na adres mail użytkownika aplikacji jest przesłana wiadomość, o wcześniej zdefiniowanej treści.

Rys.3. Widok zakładki Alarm Status

Powiadomienia komunikacją MQTT

Siec telemetryczna MQTT jest osadzona na stosie TCP i działa na podstawie wzorca publikacji-subskrypcji, w którego centrum pracuje broker MQTT. Broker pełni rolę huba sieci MQTT - klienci łączą się z brokerem w celu publikowania i odbierania wiadomości. Klienci sieci MQTT nie komunikują się ze sobą bezpośrednio i nie znają swoich numerów IP, co przekłada się na mniejszą rozpoznawalność w sieci. Dodatkowo klienci są zabezpieczeni przez uwierzytelnienie.

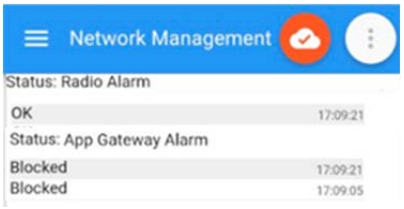
W sieci MQTT dane nie są przechowywane, a jedynie przekazywane od klienta publikującego do subskrybującego, w związku z czym broker może być implementowany na zwykłym komputerze PC.

Do wysyłania powiadomień siecią MQTT napisano oddzielny skrypt w którym utworzono instancję klienta publikującego MQTT oraz przekazano parametry połączeniowe i dane wysyłane do brokera MQTT. Posłużono się przy tym biblioteką *paho.mqtt.client* .

Podczas testów rolę brokera pełnił broker publiczny, u którego zostali zarejestrowani klienci.

Odbiór metryk stacji BSC skonfigurowano w aplikacji Android, która pełni rolę klienta subskrybującego i cyklicznie otrzymuje powiadomienia od systemu monitoringu.

Po skonfigurowaniu aplikacja zarządzająca wysyła powiadomienia do brokera, przy czym w przeciwieństwie do e-maili powiadomienia MQTT są wysyłane regularnie z informacją o wszystkich stanach alarmowych stacji (czas odświeżania można zmienić na zakładce *Notification*, domyślnie jest ustawiony na 10 sekund) (rys.4).



Rys.4. Tablica alarmów w aplikacji Android

Podsumowanie

Rozwiązania dające możliwość zdalnego monitorowania urządzeń sieciowych są niezwykle użyteczne w systemach opartych na sieci TCP/IP, w tym we współczesnych systemach radiokomunikacyjnych.

Prezentowana aplikacja zbierająca i analizująca metryki SNMP, przetestowana na systemie radiokomunikacyjnym Tetra/DMR jest w pełni funkcjonalnym narzędziem monitorowania systemów radiokomunikacji ruchomej. Zapewnia wygodny zdalny dostęp do stacji bazowej z urządzeń na platformie Windows oraz Android.

Aplikacja ma wiele zalet praktycznych: m.in. umożliwia selekcjonowanie obiektów interesujących użytkownika i przesyłanie powiadomień o ich stanie za pomocą e-maili oraz powiadomień w sieci telemetrycznej MQTT, ponadto może zostać łatwo przeniesiona na dowolny host w obrazie maszyny wirtualnej, dzięki czemu jest bezpiecznie odizolowana od komputera gospodarza.

Nowoczesne metody i narzędzia oraz funkcjonalność składają się na wartościowy produkt finalny o zastosowaniu praktycznym.

***Autorzy**: dr inż. Dorota Rabczuk, Uniwersytet Morski w Gdyni, Zakład Telekomunikacji Morskiej, ul. Morska 81-87, 81-225 Gdynia, E-mail: d.rabczuk@we.umg.edu.pl; mgr inż. Radosław Głąb, absolwent kierunku Elektronika i Telekomunikacja UMG w Gdyni, ul. Morska 81-87, 81-225 Gdynia, E-mail: radek2352@gmail.com.*

LITERATURA

[1] Mauro D., Schmidt K., Essential SNMP (reviewed), *O'REILLY*, (2016), ISBN-13: 978-0596008406

[2] Netak L., Kiwelekar A., Efficient Network Management Using SNMP, *Journal of Network and Systems Management,* 14(2) (2006), 189-194

[3] Oancea D., Structure of management information in SNMP, *The annals of DUNAREA DE JOS University of Galati*, III(2003), 84-85

[4] Hubin D., Guiyuan L., Lei Z., *Analysis and Implementation of Embedded SNMP Agent*. 4th Conference on Computer and Computing Technologies in Agriculture (CCTA), Oct 2010, Nanchang, China, 10.1007/978-3-642-18369-0_11, 96-102

[5] Kazaz T., Kulin M., Kaljić E., Čaršimanović T., One approach to the development of custom SNMP agents and integration with management systems, *Proceedings of the 35th International Convention MIPRO,* n.1 (2012), 592-596

[6] Etingof I., *PySNMP Read the Docs – Library reference*, https://pysnmp.readthedocs.io/en/latest/docs/api-reference.html, (2019), dostęp 23.03.2022