Szacowanie zajętości łącza metodą par pakietów w systemie komunikacji IoT

Agnieszka Chodorek
Politechnika Świętokrzyska, Wydział Elektrotechniki, Automatyki i Informatyki, Katedra Elektrotechniki Przemysłowej i Automatyki;
al. Tysiąclecia Państwa Polskiego 7, 25-314 Kielce

Robert R. Chodorek
AGH Akademia Górniczo-Hutnicza, Wydział Informatyki, Elektroniki i Telekomunikacji, Katedra Telekomunikacji; Al. A. Mickiewicza 30,
30-059 Kraków

Streszczenie: Jednym z problemów występujących w dużych systemach Internetu Rzeczy, złożonych z tysięcy urządzeń IoT, są przeciążenia sieci w pobliżu urządzeń pełniących rolę hubów komunikacyjnych (brokerów danych, chmur obliczeniowych). Przeciążenia te nie są w wystarczającym stopniu rozładowywane przez protokół TCP, który (ze względu na specyfikę ruchu telekomunikacyjnego w systemach IoT) nie jest w stanie prawidłowo oszacować, dostępnych dla danej transmisji, zasobów sieciowych. W artykule przedstawiono prototypowe urządzenie IoT, zbudowane na mikrokontrolerze Raspberry PI pracującym pod kontrolą systemu operacyjnego Linux, które szacuje wielkość, dostępnej dla protokołu TCP, przepustowości ścieżki komunikacyjnej. Urządzenie korzysta ze znanej metody par pakietów. Aby poprawić dokładność szacunków, użyto wariantu metody, który ocenia wielkość dostępnej przepustowości na podstawie ciągów par. Badania przeprowadzone w dedykowanej sieci lokalnej pozwoliły zarówno ocenić pracę urządzenia, jak i dokonać analizy dokładności szacunków przeprowadzanych w obecności ruchu charakterystycznego dla systemów Internetu Rzeczy. Oceniono również narzut ruchu wnoszonego do sieci Internetu Rzeczy przez pomiary metodą par testowych TCP. Ze względu na ograniczoną moc obliczeniową mikrokontrolera Raspberry PI, urządzenie korzysta z prostych, szybkich wariantów obliczeniowych metody par pakietów PTR (bez odstępu czasowego między parami pakietów) oraz zmodyfikowany IGI (ze zmiennym odstępem czasowym między parami pakietów). Urządzenie umożliwia szybką ocenę stanu sieci w trakcie trwania transmisji IoT. Znajomość stanu sieci, w tym przepustowości dostępnej dla transmisji TCP, pozwoli na efektywniejsze działanie systemu wykorzystującego dużą liczbę urządzeń Internetu Rzeczy.

Słowa kluczowe: implementacja, Internet Rzeczy, metoda par pakietów, pomiary, przeciwdziałanie przeciążeniom, Raspberry PI, protokół TCP, układ prototypowy

1. Wprowadzenie

Urządzenia Internetu Rzeczy (ang. *Internet of Things*, IoT) są obecnie wykorzystywane do różnych celów - od kontrolno-sterujących, przez szeroko rozumiany monitoring, po pomiary prowadzone na bieżąco, w czasie rzeczywistym (ang. *on-line measurements*). W systemach przemysłowych urządzenia IoT spełniają często funkcję zaawansowanych czujników pomiarowych. Podobną rolę pełnią na współczesnych statkach – zwłaszcza tych określanych mianem inteligentnego statku (ang. *smart ship*). Inteligentne statki mogą mieć na swoim pokładzie od 15 000 do 20 000 czujników [1], które monitorują zarówno statek i jego funkcje, jak i przewożony ładunek, oraz bliskie otoczenie statku (w tym warunki pogodowe). W przypadku statków autonomicznych liczba ta jeszcze wzrośnie (głównie ze względu na dodatkowe lidary, radary, kamery termowizyjne, kamery HD generujące duże ilości danych, w tym danych poddawanych lokalnej kompresji) [2].

Protokoły warstwy aplikacji dla urządzeń IoT, takie jak MQTT (ang. *Message Queue Telemetry Transport*), AMQP (ang. *Advanced Message Queuing Protocol*) oraz CoAP (ang. *Constrained Application Protocol*) typowo korzystają z usług protokołu transportowego TCP (ang. *Transmission Control*

Autor korespondujący:
Agnieszka Chodorek, a.chodorek@tu.kielce.pl

Artykuł recenzowany
nadesłany 30.05.2019 r., przyjęty do druku 28.06.2019 r.

Protocol) [3]. Mechanizm przeciwdziałania przeciążeniom, zastosowany w protokole TCP, szacuje wielkość zasobów sieciowych[1] dostępnych dla bieżącego połączenia TCP zgodnie z algorytmem AIMD (ang. *additive-increase/multiplicative-decrease*). Algorytm AIMD, arytmetycznego wzrostu (w razie braku przeciążenia) i geometrycznego spadku ilości wysyłanych danych (w razie wystąpienia przeciążenia), pozwala protokołowi elastycznie dostosowywać się do aktualnych warunków pracy sieci. Prowadzi to nie tylko do rozładowywania przeciążeń, ale i do sprawiedliwego (równomiernego) podziału zasobów sieciowych pomiędzy konkurujące połączenia TCP. Jednak, aby te efekty osiągnąć, transmisja TCP musi trwać wystarczająco długo.

Sesje TCP prowadzone na potrzeby MQTT są bardzo krótkie w porównaniu do typowej sesji TCP [4]. W artykule [5] wskazano, że metoda przeciwdziałania przeciążeniom stosowana przez TCP niewystarczająco chroni ruch IoT. Niedostateczna skuteczność dotychczasowych metod wymusiła prace nad poprawą efektywności mechanizmów przeciwdziałania przeciążeniom, w tym prac nad poprawkami do protokołu TCP [6].

W pracy [7] przedstawiono metodę przeciwdziałania przeciążeniom opartą o analizę czasów RTT (ang. *Round Trip Time*) i RTO (ang. *Retransmission Timeout*), wyznaczanych przez protokół TCP. Inna metoda, z wykorzystaniem analizy wariancji czasu RTT, została zaprezentowana w [8].

Rozwiązania szacujące przepustowość na podstawie analizy czasów RTT czy zmienności czasu RTT nie są zbyt dokładne. O wiele dokładniejsza jest metoda oparta na pomiarze czasu między parą pakietów (ang. *packet pair*) wysyłanych jeden za drugim [9, 10]. Metoda par pakietów z powodzeniem była testowana w systemach multimedialnych [11]. Modyfikacja tej metody, polegająca na wysyłaniu ciągów par pakietów PPT (ang. *Probing Packet Trains*), okazała się skuteczna w przypadku pomiaru w sieci obciążonej typowymi, długotrwałymi transmisjami TCP [10].

Metoda PPT wymaga wysyłania od kilku do nawet ponad stu ciągów po kilkadziesiąt (typowo: 60) par pakietów [10, 12]. W efekcie pojedyncze oszacowanie generuje dodatkowy, stosunkowo duży (wielokrotnie większy od całkowitego ruchu pojedynczej sesji TCP w systemie IoT) ruch pomiarowy. W systemach zawierających tysiące urządzeń IoT, tak duży ruch pomiarowy znacznie obciążałby sieć. Autorzy proponują, by pomiary metodą PPT realizowane były na potrzeby więcej niż jednego urządzenia. Wymagałoby to zbudowania autonomicznego urządzenia pomiarowego, służącego do szacowania dostępnej przepustowości na potrzeby grupy urządzeń IoT (ewentualnie brokera danych lub chmury obliczeniowej). Takie urządzenie mogłoby również przeprowadzać pomiary na potrzeby własnych transmisji IoT.

Celem prac, których podsumowaniem jest niniejszy artykuł, było praktyczne sprawdzenie skuteczności metody PPT w sieci przenoszącej ruch Internetu Rzeczy. Aby zrealizować ten cel, zaprojektowano prototypowe urządzenie na mikrokontrolerze Raspberry PI. Urządzenie zostało zbudowane i przetestowane w lokalnej sieci, w obecności generowanego ruchu IoT o dużym wolumenie.

Artykuł składa się z sześciu rozdziałów. Rozdział drugi przybliża problematykę Internetu Rzeczy w gospodarce morskiej. Rozdział trzeci prezentuje metodę par pakietów i jej wariantu PPT. Rozdział czwarty omawia prototypowy układ IoT realizujący szacowanie dostępnej przepustowości metodą par pakietów w wariancie PPT. Rozdział piąty zawiera analizę wyników zebranych w sieci gigabitowej, homogenicznej oraz heterogenicznej z pojedynczym „wąskim gardłem". Rozdział szósty stanowi podsumowanie artykułu.

[1] Głównie przepustowość łącza stanowiącego „wąskie gardło" (ang. *bottleneck*) ścieżki prowadzącej od nadajnika do odbiornika.

2. Internet Rzeczy a transport morski

Urządzenia Internetu Rzeczy dostarczają danych w całym okresie życia statku (od projektowania, przez wytwarzanie, eksploatację, aż po utylizację) [13]. W fazie projektowania wykorzystuje się dane zebrane z urządzeń IoT pracujących na statkach tego samego lub podobnego typu, a systemy komputerowego wspomagania projektowania CAD (ang. *Computer Aided Design*) mają integrować prace nad konstruowanym statkiem z nadzorem nad tworzonym oprogramowaniem IoT. W efekcie dane pozyskiwane z systemów IoT wpływają na proces projektowania i udoskonalania nie tylko statku jako takiego, ale i oprogramowania IoT.

W fazie wytwarzania – dane z urządzeń IoT zbierane są już w trakcie budowy statku. Po zamontowaniu urządzeń IoT, niektóre z nich monitorują strukturę podczas budowy [13].

W fazie eksploatacji, użycie urządzeń IoT pozwala zmniejszyć koszty eksploatacyjne i poprawić bezpieczeństwo.

Zastosowanie Internetu Rzeczy w gospodarce morskiej pozwala na poprawę efektywności transportu morskiego, a także na optymalizację kosztów spedycji dzięki zwiększeniu możliwości analizy czynników wpływających na koszty. Do czynników kosztotwórczych, których wartości są pozyskiwane na podstawie ciągłego pomiaru dokonywanego z użyciem urządzeń IoT należą, między innymi, zużycie paliwa i wody oraz aktualny stan ładunku [14]. Urządzenia IoT ograniczają koszty napraw i diagnostyki, pozwalając na monitorowanie w trybie ciągłym aktualnego stanu statku [14]. Umożliwiają zdalne sterowanie i zarządzanie urządzeniami zmniejszając konieczną obsługę na pokładzie statku.

Zastosowanie specjalizowanych urządzeń Internetu Rzeczy ma również wpływ na poprawę bezpieczeństwa transportu morskiego. Urządzenia takie dają możliwość pełnego, realizowanego w czasie rzeczywistym, monitorowania struktury statku [14, 15]. Możliwe jest monitorowanie różnego typu naprężeń i deformacji struktury statku, a także analiza wibracji, realizowana z wykorzystaniem trójosiowych czujników, umieszczonych w różnych punktach statku [15]. Systemy IoT umożliwiają monitorowanie stanu i diagnostykę urządzeń zainstalowanych na statku. Dane pozyskane z urządzeń Internetu Rzeczy pozwalają na przeprowadzenie złożonej analizy pod kątem profilaktyki urządzeń, zapobiegając awariom i zwiększając bezpieczeństwo techniczne przez działania uprzedzające.

3. Metoda par pakietów

Do szacowania dostępnej przepustowości łączy stosowane są metody pasywne i aktywne [9]. W metodach pasywnych, urządzenia komutacyjne (przełączniki, rutery) zbierają dane o ruchu w sieci. Zebrane dane są analizowane w trybie off-line. W metodach aktywnych, systemy końcowe wysyłają pakiety testujące [10]. Metody te są na tyle atrakcyjne, że wynik pomiaru jest od razu dostępny w urządzeniach końcowych. Dodatkowo, pomiar nie wymaga zbierania informacji z urządzeń sieciowych (co bywa trudne, np. ze względów bezpieczeństwa). Jedną z metod aktywnych jest metoda par pakietów. Kompleksowa analiza metod wykorzystujących parę pakietów oraz analiza metod pasywnych (obie analizy przeprowadzone zostały pod kątem stosowalności metod w środowisku sieci bezprzewodowych) zostały zaprezentowane w pracy [9].

Metoda estymacji dostępnej przepustowości za pomocą pary pakietów polega na wysłaniu dwóch kolejnych pakietów (pakiety testowe P1 i P2) z zadanym odstępem czasu między nimi. Zwykle pakiety wysyłane są jeden po drugim, tylko z przerwą technologiczną. Każdy z pakietów pary testowej jest potwierdzany przez system odbiorczy.

Między pakietami P1 i P2 nie może być nadawany żaden inny pakiet. Ewentualne pakiety, jakie pojawią się między nimi na ścieżce między nadajnikiem a odbiornikiem, a które będą wpływać na opóźnienie P2 względem P1 pochodzą zatem zawsze z zewnętrznych źródeł ruchu, rywalizujących z ruchem testowym o przepustowość.

W metodzie pary pakietów dokonuje się pomiaru lub szacowania odstępu czasu między pakietami P2 i P1:
– g_I (ang. *initial gap*) – obserwowanego podczas nadawania,
– g_B (ang. *bottleneck gap*) – obserwowanego podczas transmisji w „wąskim gardle”,
– g_O (ang. *output gap*) – obserwowanego w systemie odbiorczym.

Estymacji dostępnej przepustowości za pomocą pary pakietów dokonuje się na podstawie następującej zależności [10]:

$$g_O = g_B + \frac{B_C \cdot g_I}{B_O} \tag{1}$$

gdzie: B_C – ruch obciążający łącze w czasie transmisji pakietów P1 i P2, B_O – przepustowość „wąskiego gardła”.

W zastosowaniu praktycznym [12], autorzy [10] użyli uproszczonej i zmodyfikowanej wersji równania (1):

$$B_{AV} = \frac{L \cdot \left(2 \cdot g_I - g_O\right)}{g_I \cdot \left(\frac{L}{B_N} + g_O\right)} \tag{2}$$

gdzie: B_{AV} – dostępna przepustowość, B_N – przepustowość łącza nadawczego, L – długość pakietu P2.

Z zależności (1) lub (2) wyznaczana jest przepustowość „wąskiego gardła” ścieżki transmisyjnej między nadajnikiem a odbiornikiem. W artykule [10] zaproponowano, aby w przypadku znaczących fluktuacji czasów g_B i g_O wysyłana była nie pojedyncza para testowa, ale ciąg testowy par pakietów (wariant PPT metody par pakietów). Średnia arytmetyczna ze wszystkich wartości B_O wyznaczonych na podstawie par wchodzących w skład ciągu jest estymatorem przepustowości „wąskiego gardła”. W pracy [10] przedstawiono dwa warianty metody PPT: podstawowy PTR (ang. *Packet Transmission Rate*) oraz zmodyfikowany IGI (ang. *Initial Gap Increasing*). W wariancie IGI zwiększany jest odstęp między pakietami kolejnych, nadawanych par. Pozwala to, w pewnych warunkach, łatwiej odnaleźć nieobciążone lub słabo obciążone łącze.

4. Prototypowy układ IoT szacujący przepustowość metodą par pakietów

4.1. Wybór platformy sprzętowej

Do najpopularniejszych mikrokontrolerów stosowanych w układach IoT należą kontrolery wykorzystujące układ ESP8266. Mogą one pracować samodzielnie lub współpracować z platformą Arduino. Układ ESP8266 ma wbudowaną łączność bezprzewodową. Do Arduino współpracującego z ESP8266 można dołączyć zewnętrzny interfejs Ethernet. Urządzenia wykorzystujące układ ESP8266 mają jednak zbyt małą moc obliczeniową, aby można było uruchomić na nich algorytmy zapewniające poziom bezpieczeństwa odpowiedni dla systemów przemysłowych.

Alternatywą dla kontrolerów zbudowanych w oparciu o układ ESP8266 (lub starszy, ESP8265) są systemy wykorzystujące procesory ARM, które mają dużą moc obliczeniową przy niskim zapotrzebowaniu energetycznym. Analizując dostępne na rynku rozwiązania tej rodziny zdecydowano się na wybór mikrokontrolera Raspberry PI 3 model B+.

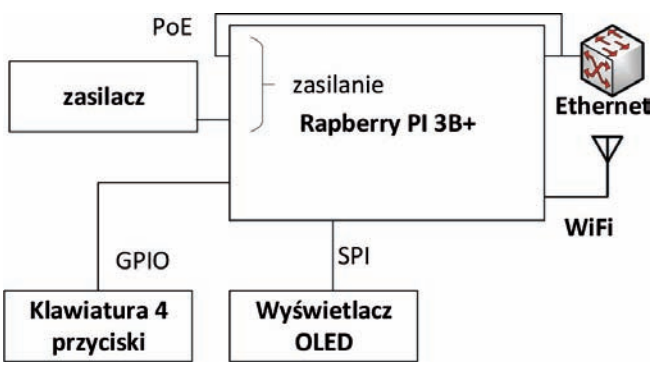
Mikrokontroler Raspberry PI 3 model B+ wykorzystuje czterordzeniowy procesor Broadcom BCM2837B0 o architekturze 64-bitowej ARM-8 Cortex-A53. Mikrokontroler ma 1 GB pamięci RAM. Jest wyposażony w trzy interfejsy sieciowe: IEEE 802.11 b/g/n/ac pracujące w pasmach 2,4 GHz i 5 GHz, Bluetooth 4.2 i Gigabit Ethernet. Ma on również szereg portów uniwersalnych, typowych dla systemów komputerowych. Są to: interfejs HDMI (ang. *High Definition Multimedia Interface*), analogowy port audio, cztery porty USB 2.0, interfejs CSI (ang. *Camera Serial Interface*) do podłączania dedykowanej kamery oraz interfejs DSI (ang. *Display Serial Interface*) do podłączenia ekranów LCD.

Z punktu widzenia pracy Raspberry PI jako urządzenia IoT istotne jest, że ma on szereg urządzeń wejścia/wyjścia GPIO (ang. *General-Purpose Input/Output*), dostępnych przez 40-stykowe złącze. Są one w pełni konfigurowalne i mogą spełniać rolę wejść lub wyjść (cyfrowych jak i analogowych). Oprócz typowego wejścia/wyjścia (przetworniki A/C C/A lub we/wy cyfrowe) możliwe jest także ustawienie wyjść:
— w trybie PWM (ang. *Pulse Width Modulation*), który pozwala na automatyczne generowanie sygnału o zadanym wypełnieniu,
— jako portów szeregowych w standardzie I²C (ang. *Inter Integrated-Circuit bus*) lub SPI (ang. *Serial Peripheral Interface*).

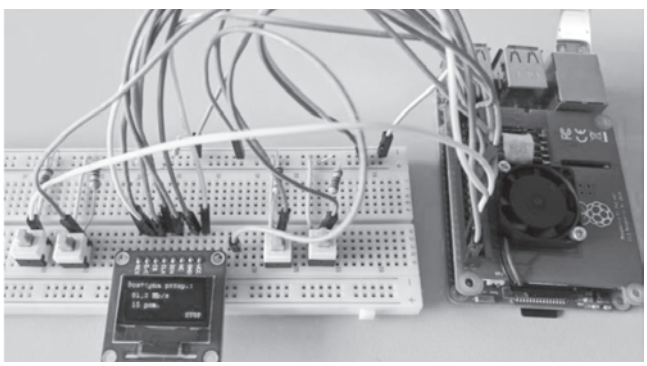
4.2. Prototyp

Schemat blokowy prototypowego urządzenia przedstawiony został na rysunku 1a. Do budowy prototypu użyto mikrokontrolera Raspberry PI 3 model B+, wyświetlacza OLED o rozdzielczości 128 px × 64 px oraz klawiatury o 4 przyciskach. Urządzenie może łączyć się przez interfejsy sieciowe

a)



b)



Rys. 1. Układ pomiarowy dla systemów IoT bazujący na Raspberry PI: a) schemat blokowy, b) rozwiązanie prototypowe (fotografia własna)
Fig. 1. Measurement device for IoT systems based on the Raspberry PI: a) block diagram, b) the prototype IoT system

Ethernet 1Gb/s i Wi-Fi. Zasilanie jest dostarczane z zewnętrznego, standardowego zasilacza lub z modułu PoE (ang. *Power over Ethernet*). Zasilanie przez sieć Ethernet ogranicza ilość niezbędnego okablowania systemu IoT. Rysunek 1b przedstawia układ z rysunku 1a, zmontowany na płytce prototypowej, podłączony do sieci Ethernet i korzystający z zasilania PoE.

Interfejs człowiek-maszyna (interfejs użytkownika) wykorzystuje prostą, czteroklawiszową klawiaturę (do sterowania urządzeniem oraz do wyboru opcji) oraz wyświetlacz OLED o rozdzielczości 128 px × 64 px (do wyprowadzania wyników i informacji pomocniczych). Klawiatura dołączona jest do wejść cyfrowych GPIO mikrokontrolera Raspberry PI. Stan wejść sprawdzany jest cykliczne. Klawisz uznawany jest za naciśniety (lub zwolniony) jeżeli stan linii powiązanej z danym klawiszem jest niezmienny przez zadany przedział czasu. Zapobiega to reakcjom na drżenie styków.

Wyświetlacz graficzny OLED [16] jest urządzeniem autonomicznym wyposażonym w układ SSD1306 obsługujący wyświetlacze OLED/PLED ze wspólną katodą o maksymalnej rozdzielczości 128 px × 64 px. Układ SSD1306 zawiera pamięć RAM obrazu. Podczas pracy nie korzysta z karty graficznej (sam stanowi niejako kartę graficzną zintegrowaną z wyświetlaczem). Pamięć obrazu jest statyczną pamięcią RAM. Każdemu bitowi pamięci odpowiada pojedynczy piksel na matrycy OLED.

Dane, które mają być wyświetlone, przesyłane są do wyświetlacza przez interfejs, który może pracować w standardzie SPI lub I²C (typ interfejsu jest wybierany przez odpowiednią kombinację dwóch zworek). Dzięki zastosowaniu techniki OLED wyświetlacz nie wymaga dodatkowego podświetlenia i jest bardzo kontrastowy. Do obsługi interfejsu wyświetlacza została zastosowana biblioteka BCM 2835 [17] (nazwa pochodzi od układu zastosowanego w Rasperry PI w wersji 3B – układ pod względem obsługi wejść i wyjść jest całkowicie zgodny z obecnie stosowanym w Rasperry PI 3B+ układem BCM2837B0). Jest to biblioteka napisana w języku C, która umożliwia odczyt i zapis wejść cyfrowych oraz magistral SPI i I²C. Ta ostatnia cecha jest istotna z punktu widzenia współpracy z wyświetlaczem.

W omawianym urządzeniu, mikrokontroler Raspberry PI pracuje w trybie tekstowym (praca w trybie graficznym powoduje znaczący wzrost zużycia energii i nagrzewanie się urządzenia). Założono, że do urządzenia nie będzie podłączany zewnętrzny monitor graficzny. Ponieważ zastosowany wyświetlacz pracuje tylko w trybie graficznym i nie korzysta z karty graficznej zintegrowanej z procesorem, do wyświetlenia tekstu konieczne były definicje map bitowych wyświetlanych czcionek. W oprogramowaniu wykorzystano definicje czcionek (o trzech różnych wielkościach) zoptymalizowanych pod kątem zastosowanego wyświetlacza OLED. Są one dostępne na stronie producenta [16].

Urządzenie pracuje pod kontrolą systemu operacyjnego Raspbian (wariant systemu Linux dla Raspberry PI) z jądrem linuksa 4.14. Podczas prac nad prototypem zostało zaprojektowane i zaimplementowane oprogramowanie, które pozwala na konfigurację urządzenia pomiarowego, określenie parametrów testów i zarządzanie pomiarem przeprowadzanym w trybie ręcznym (pojedynczego pomiaru) lub w trybie automatycznym (ciągłym). Do realizacji pomiaru metodą par pakietów zaadaptowano dostępne publicznie procedury oprogramowania [12], które mierzy dostępną przepustowość ścieżki metodą PPT z użyciem testowych ciągów par pseudopakietów (ang. *fake packets*) TCP. Procedura pomiarowa [12] jest wywoływana z poziomu oprogramowania zarządzającego procesem pomiarowym. Po przeprowadzeniu pomiaru wyniki są z niej pobierane w sposób nieobciążający mikrokontrolera Raspberry PI.

4.3. Dostosowanie procedury pomiarowych

System operacyjny Linux dostosowany do Raspberry PI ma jądro 4.14, zaś oprogramowanie [12] jest dostosowane do funkcji dostępnych przed jądrem 2.6.x. Konieczne zatem były znaczące zmiany w funkcjach ściśle współpracującym z jądrem systemu operacyjnego. Wprowadzone zmiany nie zmieniły funkcjonalnie oprogramowania [12]. Uwzględniono w nich nowe struktury danych i funkcje wykorzystywane we współczesnym jądrze systemu operacyjnego Linux. Dodatkowo Raspberry PI ma procesor z rodziny ARM, a oprogramowanie [12] tworzy pseudopakiety TCP jako strukturę, zakładając, że korzysta z procesora z rodziny x86. Różnice w wewnętrznej architekturze obu rodzin mają wpływ na integrację procedur zawartych w [12] z jądrem systemu. Jedną z cech ARM jest bowiem organizacja zapisu do pamięci pozwalająca na użycie zarówno big jak i little-endian (nazywane bi-endian). W praktyce procesor jest ustawiony w tryb little-endian (x86 ma big-endian). Wymagało to istotnych zmian w organizacji danych – rekordach (strukturach) opisujących dane oprogramowania pomiarowego.

Wprowadzono również zmiany dostosowujące oprogramowanie [12] do wymogów urządzenia pomiarowego. Po pierwsze, praca urządzenia w trybie ciągłym wymaga, by proces pobierania wyników nie zakłócał następnych pomiarów. Aby spełnić to wymaganie, należało zmienić sposób pobierania wyników użyty w [12], który pozwalał jedynie na pracę w trybie pojedynczego, zadawanego ręcznie pomiaru. Wyniki obecnie są odczytywane, z poziomu użytkownika, przez zmienne systemowe.

Inne wprowadzone zmiany to m.in. dodanie odczytu aktualnych parametrów sieci macierzystej urządzenia oraz całkowita zmiana koncepcji oprogramowania pomiarowego (z aplikacji użytkownika na usługę w systemie operacyjnym).

4.4. Oprogramowanie pomiarowe jako usługa w systemie operacyjnym

Oryginalne oprogramowanie [14] współpracuje tylko z terminalem, co w praktyce oznacza, że aby uruchomić pomiar należałoby zalogować się na mikrokontrolerze Raspberry PI i wydać odpowiednie polecenie. Tymczasem nasza koncepcja zakładała, że oprogramowanie pomiarowe powinno być uruchamiane automatycznie w odpowiedniej fazie uruchamiania systemu, i to niezależnie od tego, czy użytkownik się zaloguje na mikrokontrolerze, czy nie. Aby ten cel osiągnąć, należy uruchamiać oprogramowanie pomiarowe jako usługę systemu operacyjnego Linux.

Najogólniej rzecz ujmując, usługa to aplikacja „obudowana" odpowiednimi skryptami i uruchamiana przez system operacyjny. Aby była to pełnoprawna usługa systemu Linux, wymagane jest zdefiniowanie skryptu startowego (w przypadku naszego urządzenia utworzono skrypt o nazwie `iot_tests`), który jest zapisany w katalogu /etc/init.d. Skrypt taki powinien zawierać zdefiniowane akcje dla trzech podstawowych zadań (start, stop i restart), realizowanych przez każdą z usług w systemie operacyjnym Linux:

- uruchomienia usługi (start),
- zatrzymania usługi (stop),
- przeładowania oprogramowania usługi (reload).

Identyfikatory zadań są przekazywane jako parametry do skryptu startowego, który, w odpowiedzi, uruchamia oprogramowanie usługi z określonymi parametrami.

Uruchomienie oprogramowania (usługi) podczas startu wymaga zarejestrowania skryptu (tu: `iot_tests`) jako uruchamianego podczas startu systemu (rejestracja dokonywana jest komendą: `update-rc.d iot_tests defaults`). Usługa pomiarowa jest uruchamiana w ostatniej fazie uruchamiania systemu, kiedy już działają wszystkie inne usługi systemowe (w tym podstawowa obsługa interfejsów sieciowych).

4.5. Uruchamianie urządzenia i przeprowadzanie pomiarów

Oprogramowanie urządzenia pomiarowego jest uruchamiane podczas startu systemu operacyjnego, jako usługa w systemie (wariant domyślny). Dzięki temu urządzenie od razu pracuje autonomicznie, korzystając ze swoich własnych interfejsów użytkownika. Tuż po uruchomieniu, oprogramowanie odczytuje dane z pliku konfiguracyjnego, w którym są zapisywane podstawowe informacje niezbędne do przeprowadzania pomiarów. Ma on format pliku tekstowego z sekcjami odpowiadającymi poszczególnym danym konfiguracyjnym. Jeżeli plik konfiguracyjny nie istnieje, oprogramowanie tworzy taki plik, a następnie przechodzi do procedury ustawiania parametrów konfiguracyjnych. Jeżeli plik istnieje, oprogramowanie od razu przechodzi do podstawowego menu.

Menu urządzenia jest pokazywane na dole wyświetlacza. Wybór opcji odbywa się za pomocą klawiszy (na rysunku 1b są to klawisze, które na płytce prototypowej znajdują się najbliżej wyświetlacza). Podstawowe menu pozwala na uruchomienie pomiaru zajętości łącza (opcja „*POMIAR*") oraz ustawienie lub zmianę istniejących ustawień parametrów konfiguracyjnych pomiarów (opcja „*KONFIGURACJA*").

Jeżeli na wyświetlaczu pojawia się opcja „*POMIAR*", urządzenie jest gotowe do wykonywania pomiarów. Urządzenie ma możliwość wykonywania pomiaru pojedynczego bądź ciągłego dla wskazanych: interfejsu wyjściowego i systemu docelowego. Podczas pojedynczego pomiaru do systemu docelowego wysyłanych jest, przez wskazany interfejs, k ciągów po n par pseudopakietów TCP. Wartość k zależy od stanu sieci i dobierana jest automatycznie przez procedury oprogramowania [12]. System docelowy musi mieć uruchomioną aplikację echa, która odbiera pseudopakiet, pozbawia go pola danych i wysyła sam nagłówek do urządzenia pomiarowego. Urządzenie dokonuje pomiaru czasu, jaki upłynął między odbiorem nagłówków należących do tej samej pary pseudopakietów. Czas ten stanowi podstawę do obliczeń dostępnej przepływności ścieżki między urządzeniem pomiarowym a systemem docelowym. Jako aplikację echa wykorzystano aplikację serwera (ang. *server*) z oprogramowania [12], ze zmianami dostosowującymi ją do jądra 4.14 systemu operacyjnego Linux.

Po wybraniu opcji „*POMIAR*" rozpoczyna się wykonywanie pomiaru, jednorazowego na żądanie (w trybie pomiaru pojedynczego) lub wielokrotnego (w trybie pomiaru ciągłego). Pomiar wielokrotny jest realizowany jako seria pomiarów pojedynczych, wykonywanych w pętli nieskończonej zgodnie z zadanymi w konfiguracji parametrami. Użytkownik może w dowolnej chwili zatrzymać pomiar wielokrotny naciskając klawisz odpowiadający widocznemu na wyświetlaczu napisowi „*STOP*". Po każdym pojedynczym pomiarze wyświetlane są: dostępna przepustowość łącza i liczba k ciągów par składających się na pomiar. Aby uniknąć niejednoznaczności, oprogramowanie wyświetlające wyniki odczytuje stan zmiennych systemowych podających dostępną przepustowość łącza dopiero wówczas, gdy zostanie zakończona procedura pomiarowa.

4.6. Konfigurowanie urządzenia

Konfigurowanie urządzenia pomiarowego może odbywać się z menu (opcja „*KONFIGURACJA*") lub, po zalogowaniu przez terminal tekstowy (ssh), przez ręczną zmianę wpisów do pliku konfiguracyjnego (w dowolnym edytorze tekstowym, dostępnym w mikrokontrolerze). Druga możliwość nie wynika z podstawowej funkcji urządzenia pomiarowego, ale z zastosowania Raspberry PI – uniwersalnego mikrokontrolera z systemem Linux. Konfigurowanie odbywa się wówczas poza oprogramowaniem narzędziowym urządzenia pomiarowego i konieczny jest restart usługi (realizowany przez wydanie komendy: `service iot_tests reload`).

Konfigurowanie urządzenia ma na celu zdefiniowanie systemów docelowych, stanowiących koniec testowanej ścieżki, oraz ustawienie parametrów testów ścieżki. Przyjęto założenie, że każdy system docelowy będzie miał lokalną etykietę, skojarzoną z jego adresem IP. Etykiety umożliwiają nadawanie systemom docelowym, identyfikowanym przez adresy IP, nazw przyjaznych dla użytkownika. Jeżeli w sieci istnieją nazwy przypisane przez system DNS (ang. *Domain Names System*), i są one widoczne przez usługę *Reverse DNS*, pojawią się one obok etykiety nadanej na etapie konfiguracji. Ponieważ nazwy zdefiniowane w DNS zazwyczaj są długie, a wyświetlacz ma ograniczone pole wyświetlania, etykiety są podstawową formą identyfikacji systemu docelowego.

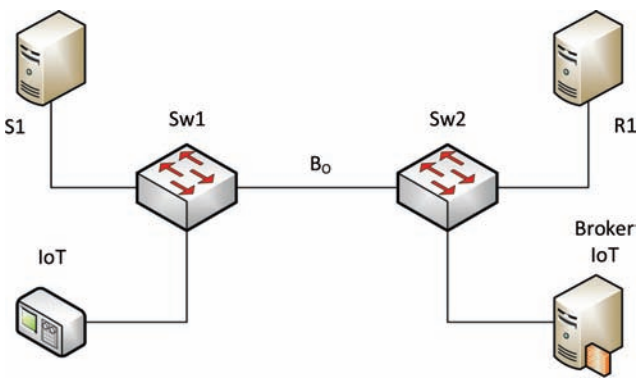
Podczas konfigurowania parametrów testów ścieżki wybierany jest interfejs sieciowy (Ethernet 1 Gb/s, Wi-Fi lub opcjonalne urządzenia sieciowe podłączone łączem USB), za pomocą którego będzie przeprowadzany pomiar. Interfejs ten jest początkiem ścieżki prowadzącej do systemu docelowego. Podczas konfiguracji wybierany jest również tryb pomiaru (ciągły, pojedynczy), wariant metody PPT (PTR lub IGI), a także długość pojedynczego ciągu testowego, mierzona liczbą par pakietów n wysyłanych w jednym ciągu testowym (wartość n ustawiana jest domyślnie na 60).

Prawidłowo skonfigurowane urządzenie jest gotowe do wykonywania pomiarów (na wyświetlaczu pojawia się wówczas opcja „*POMIAR*"). W przypadku braku danych koniecznych do rozpoczęcia pomiaru, procedura pomiarowa nie może być uruchomiona i nie jest wyświetlana opcja „*POMIAR*" (wyświetlana jest tylko opcja „*KONFIGURACJA*").

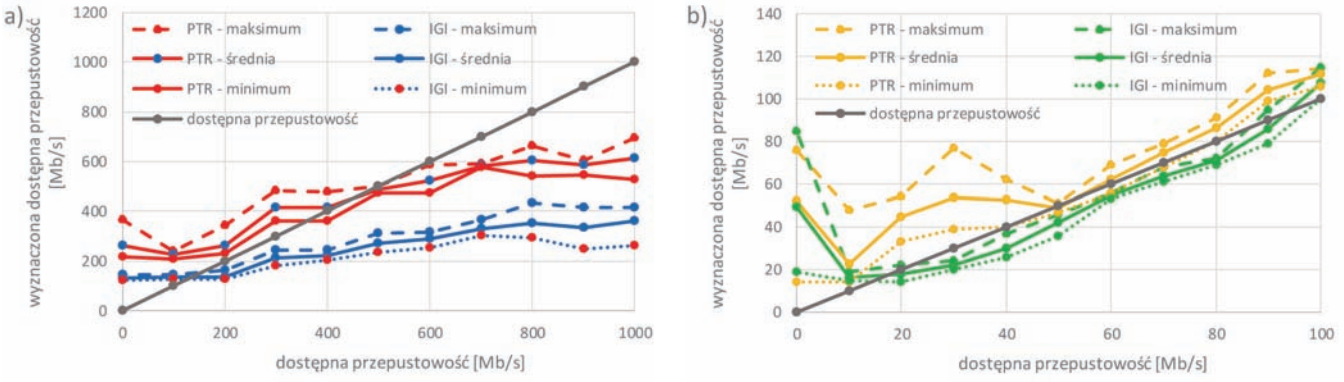
5. Wyniki

Przedstawiony w poprzednim rozdziale układ pomiarowy został przetestowany w sieci laboratoryjnej o typowej topologii pojedynczego „wąskiego gardła", przedstawionej na rys. 2. Przepustowość B_O łącza między przełącznikami Sw1 i Sw2 wynosiła 1 Gb/s (brak „wąskiego gardła") lub 100 Mb/s („wąskie gardło"). Pozostałe łącza miały przepustowość 1 Gb/s.

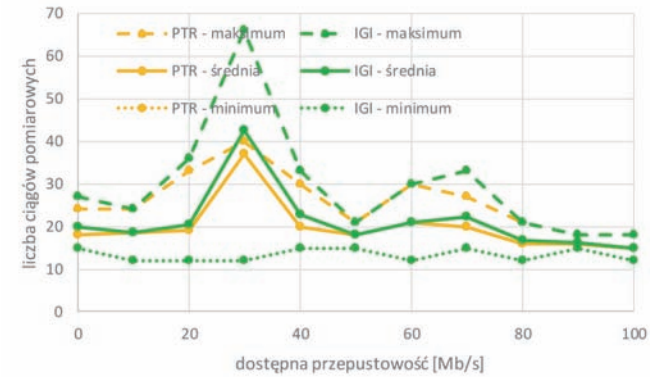
Między nadajnikiem S1 a odbiornikiem R1 przesyłany był ruch podkładowy, zadawany z gradacją co 10% przepustowości B_O. Jako generatora ruchu użyto oprogramowania MQTTBox [18], które generuje sztuczny ruch IoT (ruch protokołu MQTT pracującego nad protokołem TCP) w postaci krótkich, słabo sterowalnych sesji TCP. Aby wygenerować każde 100 Mb/s trzeba było użyć średnio ponad 3700 jednocześnie otwartych sesji TCP.



Rys. 2. Topologia testowa
Fig. 2. Test topology



Rys. 3. Estymacja dostępnej przepustowości: a) B_O = 1 Gb/s, b) B_O = 100 Mb/s
Fig. 3. The estimation of available bandwidth: a) B_O = 1 Gbps, b) B_O = 100 Mbps



Rys. 4. Podstawowe statystyki ciągów pomiarowych (B_O = 100 Mb/s)
Fig. 4. Basic statistics of probing packet trains (B_O = 100 Mbps)

Pomiar dostępnej przepływności odbywa się na ścieżce między urządzeniem pomiarowym (IoT) a brokerem IoT, na którym uruchomiona została aplikacja echa. Pomiary wykonywane były z użyciem stałej (zerowej – metoda PTR) lub zmiennej (metoda IGI) przerwy między pseudopakietami TCP wchodzącymi w skład pary testowej. Każda seria pomiarowa wykonywana była dziesięciokrotnie. Wyniki, uśrednione po 10 pomiarach, zostały zobrazowane na rysunkach 3 i 4.

Rysunek 3 przedstawia wykresy wyznaczonej dostępnej przepustowości w funkcji dostępnej przepustowości (różnicy między B_O a zadaną przepływnością ruchu podkładowego). Odchylenie wykresów od czarnej linii, obrazującej idealne oszacowanie, ma sens błędu bezwzględnego. Jak widać na rysunku, metoda par pakietów lepiej się sprawdza w przypadku niższych B_O, co jest zjawiskiem opisywanym w literaturze [9]. W przypadku ruchu IoT metoda PTR daje wyższe szacunki niż metoda IGI. Dla B_O równego 100 Mb/s, w zakresie od 20 do 90 Mb/s dostępnej przepustowości, PTR przeszacowywała, a IGI niedoszacowywała dostępną przepustowość ścieżki. Dla B_O równego 1 Gb/s, kąt nachylenia krzywych przepływności wyznaczonych obiema metodami jest zbyt mały, by zjawisko to wystąpiło w szerszym zakresie. Warto zauważyć, że w przypadku dużej zajętości łącza stanowiącego „wąskie gardło" (a tym samym małej dostępnej przepustowości) obie metody znacznie przeszacowują dostępną przepustowość, co należy uwzględnić przy projektowaniu mechanizmu przeciwdziałania przeciążeniom.

W przypadku $B_O = 1$ Gb/s, pomiar zawsze odbywał się z wykorzystaniem $k = 15$ ciągów par pakietów. W przypadku $B_O = 100$ Mb/s, liczba k była zmienna i wahała się od ponad 10 do blisko 70 (rys. 4). Większy rozrzut k dawała zawsze metoda IGI, mniejszy PTR. W obu przypadkach najmniejszy rozrzut k zaobserwowano, gdy łącze 100 Mb/s było obciążone w 50%. Wartości średnie k, wyznaczone dla obu metod, są do siebie zbliżone lub sobie równe.

6. Zakończenie

W artykule zostało przedstawione prototypowe urządzenie IoT, którego zadaniem jest szacowanie przepustowości, dostępnej dla ruchu IoT, na określonej ścieżce w sieci. Urządzenie zostało zbudowane przy wykorzystaniu mikrokontrolera Raspberry PI pracującego pod kontrolą systemu operacyjnego Linux. Zaprezentowane urządzenie korzysta z metody par pakietów w wersji TTR, z użyciem metod PTR i IGI. Urządzenie zostało przebadane w sieci testowej. Oszacowanie dostępnej przepustowości dla ruchu IoT pozwoli na zwiększenie efektywności funkcjonowania systemu, w którym równocześnie pracuje kilka tysięcy lub więcej urządzeń IoT.

Bibliografia

1. Kobyliński L., *Smart ships – autonomous or remote controlled?*, „Zeszyty Naukowe Akademii Morskiej w Szczecinie", Nr 53 (125), 2018, 28–34.
2. Höyhtyä M., Huusko J., Kiviranta M., Solberg K., Rokka J., *Connectivity for autonomous ships: Architecture, use cases, and research challenges*, 2017 International Conference on Information and Communication Technology Convergence (ICTC), October 2017, Jeju, 345–350, DOI: 10.1109/ICTC.2017.8191000.
3. Rayes A., Salam S., *Internet of things—from hype to reality*, „The road to Digitization", Springer International Publishing, Cham, Switzerland, 2019.
4. Chodorek A., Chodorek R.R., *O gwarantowanej jakości usługi Internetu Rzeczy zintegrowanej z systemem wideokonferencyjnym*, „Studia Informatica", Vol. 38, No. 3, 2017, 167–176.
5. Mishra N., Verma L.P., Srivastava P.K., Gupta A., *An analysis of IoT congestion control policies*, „Procedia computer science", 132, 2018, 444–450, DOI: 10.1016/j.procs.2018.05.158.
6. Kafi M.A., Djenouri D., Ben-Othman J., Badach, N., *Congestion control protocols in wireless sensor networks: A survey*, „IEEE communications surveys & tutorials", Vol. 16, No. 3, 2014, 1369–1390, DOI: 10.1109/SURV.2014.021714.00123.
7. Bhalerao R., Subramanian S.S., Pasquale J., *An analysis and improvement of congestion control in the CoAP*

Internet-of-Things protocol, 13th IEEE Annual Consumer Communications & Networking Conference, (CCNC), 9–12 Jan. 2016, Las Vegas, 889–894, DOI: 10.1109/CCNC.2016.7444906.
8. Betzler A., Gomez C., Demirkol I., Paradells J., *CoAP congestion control for the internet of things*, „IEEE Communications Magazine", Vol. 54, No. 7, 2016, 154–160, DOI: 10.1109/MCOM.2016.7509394.
9. Mukta M., Gupta N., *Bandwidth Estimation Tools and Techniques: A Review*, „International Journal of Research", Vol. 4, No. 13, 2017, 1250–1265.
10. Hu N., Steenkiste P., *Evaluation and characterization of available bandwidth probing techniques*, „IEEE journal on Selected Areas in Communications", Vol. 21, No. 6, 2003, 879–894, DOI: 10.1109/JSAC.2003.814505.
11. Liu Q., Hwang J.N., *A new congestion control algorithm for layered multicast in heterogeneous multimedia dissemination*, 2003 International Conference on Multimedia and Expo (ICME'03), 6–9 July 2003, Baltimore, MD, USA, Vol. 2, II-533, DOI: 10.1109/ICME.2003.1221671.
12. IGI/PTR, url: [www.cs.cmu.edu/~hnn/igi] (dostęp: 14 maja 2019).
13. Muñoz J.A., Pérez R., *Design of smart ships for the IoT*, Second International Conference on Internet of things, Data and Cloud Computing, ACM, ICC '17, March 22-23 2017, Cambridge, United Kingdom, Article No. 35, 1–7, DOI: 10.1145/3018896.3018930.
14. Perera L.P., Handling big data in ship performance and navigation monitoring, „Smart Ship Technology", 2017, 89–97.
15. Yang S., Shi L., Chen D., Dong Y., Hu Z. *Development of ship structure health monitoring system based on IOT technology*, IOP Conference Series: Earth and Environmental Science, IOP Publishing, Vol. 69, No. 1, 2017, 1–6, DOI: 10.1088/1755-1315/69/1/012178.
16. 0.96inch OLED (B), url: [www.waveshare.com/wiki/0.96inch_OLED_(B)] (dostęp: 14 maja 2019).
17. bcm2835, url: [www.airspayce.com/mikem/bcm2835] (dostęp: 14 maja 2019).
18. MQTTBox, url: [http://workswithweb.com/mqttbox.html] (dostęp: 14 maja 2019).

Estimation of Link Occupancy During TCP Transmissions in Internet of Things

Abstract: One of the serious problems with large-scale Internet of Things systems, composed of thousands of IoT devices, are network congestions that occur near communication hubs (data brokers, computing clouds). These congestions cannot be enoughly discharged by the TCP protocol, which (due to specific teletraffic, generated by IoT devices) is not able to correctly estimate bandwidth available for a given transmission. In this article, a prototype IoT device that estimates amount of bandwidth of transmission path, available for TCP transmissions, is presented. The device is built with the use of the Raspberry PI microcontroller, working under the control of the Linux operating system, and uses packet pairs method for bandwidth estimation. To improve estimation accuracy, Probing Packet Trains (PPT) variant of packet pairs method was used. Results of experiments carried out in local area network are presented in figures and includes both analysis of estimation accuracy, and analysis of amount of control traffic that will be injected to an IoT network during a single measurement with the use of several probing packet trains. Due to limited computing power of the Raspberry PI, the device uses two, simple for computing, versions of the PPT: Packet Transmission Rate and Initial Gap Increasing. The device enables fast assessment of networks conditions. Knowledge of bit rate available for current TCP transmissions allows for more efficient performance of IoT systems that use large amount of devices.

Keywords: congestion control, implementation, Internet of Things, measurements packet pair, prototype device, Raspberry PI, TCP

dr inż. Agnieszka Chodorek
a.chodorek@tu.kielce.pl

Absolwentka Wydziału Elektrotechniki i Automatyki Politechniki Świętokrzyskiej (1991 r.). Stopień doktora nauk technicznych uzyskała na Wydziale Elektrotechniki, Automatyki, Informatyki i Elektroniki Akademii Górniczo Hutniczej (2001 r.). Zatrudniona jest obecnie na stanowisku wykładowcy w Katedrze Elektrotechniki Przemysłowej i Automatyki na Politechnice Świętokrzyskiej w Kielcach. Jej zainteresowania badawcze obejmują sieci mobilne i bezprzewodowe, przesyłanie wideo w sieci, a także protokoły transportowe i routingu. Jest autorką lub współautorką około 90 prac naukowych i dwóch książek. Członek Polskiego Towarzystwa Elektrotechniki Teoretycznej i Stosowanej, IEEE (Senior Member), IEEE Communications Society, IEEE Computer Society, IEEE Industrial Electronics Society oraz kilku branżowych IEEE Community.

dr inż. Robert Ryszard Chodorek
chodorek@kt.agh.edu.pl

Absolwent Wydziału Elektrotechniki i Automatyki Politechniki Świętokrzyskiej (1990 r.). Stopień doktora nauk technicznych uzyskał na Wydziale Elektrotechniki, Automatyki i Elektroniki Akademii Górniczo-Hutniczej (1996 r.). Zatrudniony jest obecnie na stanowisku adiunkta w Katedrze Telekomunikacji AGH. Specjalizuje się w analizie i modelowaniu protokołów sieci teleinformatycznych, sieci szerokopasmowych, transmisji multikastowej i multimedialnej. Autor i współautor ponad 100 publikacji naukowych (w tym 3 książek) z zakresu protokołów komunikacyjnych i sieci komputerowych. Członek Polskiego Towarzystwa Elektrotechniki Teoretycznej i Stosowanej, IEEE (Senior Member), IEEE Communications Society, IEEE Computer Society, IEEE Industrial Electronics Society oraz kilku branżowych IEEE Community.