PRZEGLĄD TELEINFORMATYCZNY
NR 00, 2023

Bezpieczna i wydajna wymiana danych w sieciach Internetu Rzeczy – przegląd technologii

Sebastian ŁESKA

Instytut Teleinformatyki i Cyberbezpieczeństwa, Wydział Cybernetyki, WAT,
ul. gen. Sylwestra Kaliskiego 2, 00-908 Warszawa
sebastian.leska@wat.edu.pl

STRESZCZENIE: Urządzeniom IoT często towarzyszą liczne ograniczenia sprzętowe, takie jak ograniczona moc obliczeniowa, ograniczone zasoby pamięciowe oraz niska przepustowość sieci. Wszystkie te składową mają znaczący wpływ na trudności w zapewnieniu bezpieczeństwa sieciom IoT. Z tego powodu powstały różne rozwiązania przeznaczone specjalnie dla urządzeń IoT, mające na celu podniesienie poziomu bezpieczeństwa takich urządzeń. W artykule dokonano przeglądu najpopularniejszych rozwiązań wykorzystywanych w sieciach IoT, które minimalizują problemy spowodowane licznymi ograniczeniami.

SŁOWA KLUCZOWE: IoT, bezpieczeństwo sieci IoT, dystrybucja kluczy symetrycznych, lekkie protokoły wymiany danych, interfejsy komunikacyjne IoT

Wprowadzenie

Początek XXI wieku to czas, w którym wiele badań i pracy przeznaczonych zostało na rozwój i wzrost liczby zastosowań technologii Internetu Rzeczy (ang. *Internet of Things* – IoT). Dzięki dynamicznemu rozwojowi technologii IoT możliwe jest coraz szersze wykorzystywanie oferowanych przez nie usług oraz wdrażanie nowych rozwiązań w różnych obszarach życia i pracy. Masowa produkcja urządzeń IoT wymaga jednak, aby koszt takich urządzeń był niewielki, a to oznacza, że producenci narzucają duże ograniczenia związane m.in. z mocą obliczeniową, zasobami pamięciowymi i prędkością transmisji danych. Szybki rozwój oraz tania produkcja urządzeń, a co za tym idzie, niewystarczająca dbałość o bezpieczeństwo, są przyczyną nieustannie rosnącej liczby zagrożeń oraz punktów ataku na sieci IoT ze względu na niewystarczające zabezpieczenia

urządzeń. W tym celu konieczne jest opracowanie rozwiązania zapewniającego bezpieczną wymianę danych oraz budowę zaufanego środowiska IoT.

1. Budowa zaufanego środowiska dla sieci IoT

Jednym z najistotniejszych problemów związanych z projektowaniem sieci IoT jest zapewnienie zaufania pomiędzy węzłami sieci. Każdy z węzłów powinien mieć pewność dotyczącą tożsamości i wiarygodności pozostałych stron biorących udział w wymianie danych. Zaufanie pomiędzy węzłami sieci można osiągnąć poprzez implementację mechanizmu uwierzytelniania.

1.1. Kryptografia asymetryczna

W tradycyjnych sieciach wykorzystujących protokół IP w celu zweryfikowania innego węzła stosuje się certyfikaty i podpisy cyfrowe. Jest to najbezpieczniejszy sposób uwierzytelniania, jednakże jego zastosowanie nie zawsze jest możliwe w sieciach IoT.

Do stosowania certyfikatów konieczna jest obecność zaufanej trzeciej strony biorącej udział w komunikacji pomiędzy dwoma węzłami, której rolę pełni Urząd Certyfikujący (ang. *Certificate Authority* – CA). Zadaniem CA jest weryfikacja danych zawartych w wysłanym żądaniu CSR (ang. *Certificate Signing Request*), do których zalicza się klucz publiczny oraz informacje o stronie aplikującej (np. domena, e-mail). W przypadku stwierdzenia poprawności otrzymanych danych, CA potwierdza tożsamość aplikanta poprzez wystawienie certyfikatu (podpisanie cyfrowe żądania CSR) oraz odesłanie go z powrotem. Certyfikat taki może zostać wysłany do dowolnego podmiotu w sieci, a jego obecność świadczy o wiarygodności nadawcy, zapewniając odbiorcę, że jego rozmówca jest tym, za kogo się podaje. Rozwiązanie to jest często deklasowane w sieciach IoT z powodu ich częstej niezależności od sieci Internet, a tym samym braku dostępu do istniejących CA.

Certyfikacja i podpisy cyfrowe są elementem kryptografii asymetrycznej, co oznacza, że konieczne jest stosowanie przez węzły sieci kluczy prywatnych i publicznych. Szyfrowanie asymetryczne standardowo wykorzystuje klucze o długości 2048 bitów, natomiast szyfrowanie symetryczne zazwyczaj stosuje klucze o długości 256 bitów. Oznacza to, że do obsługi kryptografii asymetrycznej konieczne jest dysponowanie większymi zasobami pamięciowymi oraz większą przepustowością sieci. W przypadku węzłów IoT często może brakować miejsca w pamięci na przechowywanie wielu takich kluczy, co uniemożliwiłoby im komunikację z częścią węzłów sieci. Dodatkowo w sieciach IoT często stosuje się interfejsy sieciowe zaprojektowane specjalnie dla urządzeń IoT, umożliwiające

bezprzewodową transmisję danych na duże odległości, jednakże z niewielką przepustowością, co oznacza, że konieczne jest ograniczenie liczby transmitowanych bajtów do minimum w celu zapewnienia wydajnej komunikacji.

1.2. Uwierzytelnianie z wykorzystaniem hasła

Jedną z najprostszych metod uwierzytelniania jest implementacja obsługi hasła. Użytkownik pragnący uzyskać dostęp do określonych zasobów bądź usług zobowiązany jest podać ustalone wcześniej hasło, które będzie świadczyło o jego wiarygodności. Hasło to wprowadza się jako ciąg znaków ASCII, który następnie poddawany jest obróbce funkcji skrótu i wysyłany do serwera w celu porównania otrzymanego skrótu z zapisanym w lokalnych zasobach skrótem. W przypadku gdy serwer stwierdzi, że hasło jest prawidłowe, wygenerowany zostanie dla klienta unikatowy token, który może być wykorzystywany do uwierzytelniania i autoryzacji bez konieczności ciągłego podawania hasła.

Metoda z wykorzystaniem hasła jest prosta w implementacji i szybka w obsłudze, a do tego nie obciąża zasobów pamięciowych ani łącza danych. Wadą jest fakt, że opcja ta nie jest tak bezpieczna, jak wykorzystanie kryptografii asymetrycznej. W tym przypadku konieczne jest stosowanie bezpiecznego hasła, które będzie wymieniane co pewien czas. Słabe hasło szybko mogłoby zostać złamane, dając nieautoryzowany dostęp do zasobów niepożądanym podmiotom. Dodatkowo stosowanie hasła wymaga, aby urządzenia były obsługiwane przez ludzi, którzy te hasła by wprowadzali, a w sieciach przemysłowych IoT konieczna jest pełna automatyzacja niewymagająca interwencji człowieka.

1.3. Uwierzytelnianie z wykorzystaniem kontekstu

Jedną z mniej znanych metod uwierzytelniania jest metoda wykorzystująca kontekst urządzenia. Rozwiązanie to polega na użyciu unikatowych cech urządzenia w procesie potwierdzania swojej tożsamości. Kontekst może składać się z wielu cech urządzenia, a każdą cechą może być charakterystyczna informacja, począwszy od uruchomionego systemu, numerów seryjnych, a skończywszy na pomiarach napięć na płytce. Zestaw takich cech tworzy kontekst, który jest w stanie jednoznacznie odróżnić jedno urządzenie od drugiego, tym samym umożliwiając jego uwierzytelnienie.

Rozwiązanie to jest proste w implementacji i umożliwia łatwą automatyzację pracy sieci, gdyż nie jest konieczna ingerencja człowieka do uwierzytelniania węzłów sieci. Metoda ta ma jednak jeden mankament, a mianowicie w przypadku dokonania niewielkiej zmiany na urządzeniu nie będzie możliwe dalsze uwierzytelnianie tego urządzenia w sieci i w takiej sytuacji konieczne będzie podjęcie stosownych działań. Z tego powodu niezwykle ważne

jest, aby dobrać taki zestaw cech, dzięki któremu możliwe będzie prawidłowe uwierzytelnienie urządzenia z równoczesnym minimalizowaniem ryzyka zmiany tych cech w czasie eksploatacji urządzenia.

2. Zapewnienie bezpiecznej transmisji danych z zachowaniem skalowalności sieci

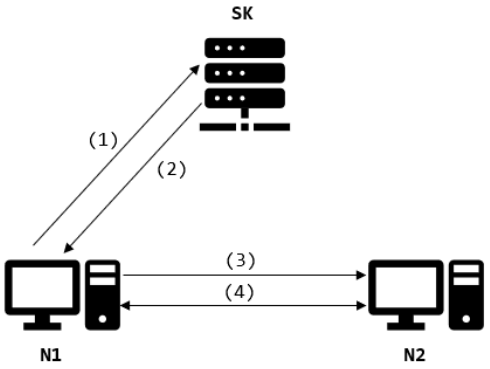
Ze względu na liczne ograniczenia urządzeń IoT, często niemożliwe jest stosowanie kryptografii asymetrycznej do zabezpieczenia transmisji danych. Oznacza to, że konieczne jest stosowanie kryptografii symetrycznej. Niesie to ze sobą jednak pewien problem, którym jest dystrybucja klucza symetrycznego. W przypadku nawiązania komunikacji pomiędzy dwoma węzłami, sprawa jest prosta, ponieważ można wykorzystać jeden z protokołów ustalania klucza wspólnego, do których należy protokół Diffiego-Hellmana bądź ECDH (ang. *Eliptic Curve Diffie-Hellman*). Sprawa jednak się komplikuje w momencie w którym konieczne jest zapewnienie skalowalności sieci, w której grupa węzłów będzie mogła stosować jeden klucz do zabezpieczenia transmisji danych. W takiej sytuacji konieczne jest wdrożenie protokołu dystrybucji klucza.

2.1. Key Distribution Center

Jednym z pierwszych rozwiązań umożliwiających dystrybucję klucza do wielu węzłów jest protokół Needham-Schroeder [1] opracowany w 1978 r. Rozwiązanie zakłada istnienie w sieci węzła pełniącego funkcję Centrum Dystrybucji Kluczy (ang. *Key Distributrion Center* – KDC) i odpowiedzialnego za generowanie i dystrybucję kluczy symetrycznych (sesji) na żądanie. Zgodnie z tą koncepcją, klucze są generowane w węźle centralnym, a dopiero potem wysyłane do poszczególnych węzłów. W celu umożliwienia takiego rozwiązania konieczne jest wcześniejsze ustalenie przez każdy dołączający do sieci węzeł klucza głównego (np. z wykorzystaniem protokołu Diffiego-Hellmana) z węzłem centralnym. Klucz ten wykorzystywany jest do zabezpieczenia dalszej transmisji danych pomiędzy tymi dwoma węzłami, w tym do zaszyfrowania transmitowanego klucza sesji. Ważne jest, aby ze względów kryptoanalitycznych klucz główny miał większą długość niż klucz sesji.

Po dołączeniu do sieci każdego klienta (Ni), węzeł centralny pełni funkcję serwera kluczy (SK) i przechowuje w swoich zasobach klucze główne wszystkich węzłów sieci. Po ustaleniu kluczy głównych dystrybucja klucza sesji odbywa się w czterech krokach (rys. 1):

1) **E(K_{N1}, R) –** wysłanie przez węzeł N1 do węzła SK żądania (R) wygenerowania klucza symetrycznego do komunikacji z węzłem N2, zaszyfrowanego kluczem głównym węzła N1.
2) **E(K_{N1}, E(K_{N2}, K_S)+K_S) –** wysłanie przez SK odpowiedzi składającej się z paczki dla węzła N2, zaszyfrowanej jego kluczem głównym oraz zawierającej klucz sesji K_S, a także paczki dla węzła N1, dołączonej na koniec paczki węzła N2, całość zaszyfrowana kluczem głównym węzła N1.
3) **E(K_{N2}, K_S) –** przesłanie do węzła N2 paczki przygotowanej przez SK, otrzymanej w wyniku odszyfrowania odpowiedzi serwera przez węzeł N1.
4) Testowanie przez węzły N1 i N2 nowo otrzymanego klucza Ks.



Rys. 1. Schemat dystrybucji klucza metodą KDC

2.2. Key Translation Center

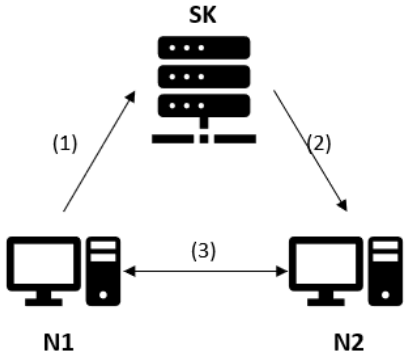
Stosowanie KDC daje gwarancję dotyczącą wiarygodności wygenerowanych kluczy dzięki pewności co do zastosowanego źródła entropii oraz wykorzystanych procedur. Rozwiązanie to jednak może być czasochłonne ze względu na złożoność obliczeniową funkcji generowania kluczy. Oznacza to, że przy wielu równoczesnych żądaniach wygenerowania klucza węzeł centralny nie będzie w stanie odpowiedzieć każdemu klientowi na czas. Z tego powodu powstała nowa metoda dystrybucji kluczy, polegająca na zastosowaniu Centrum Tłumaczenia Kluczy (ang. *Key Translation Center* – KTC) [2].

Rozwiązanie to, podobnie jak KDC, wymaga ustalenia z każdym klientem klucza głównego, jednakże w tym przypadku węzeł centralny nie generuje samodzielnie kluczy sesji, a jedynie odpowiada za ich odszyfrowanie kluczem głównym jednego węzła i zaszyfrowanie kluczem głównym drugiego węzła. Klucze sesji generowane są przez klienta i wysyłane do KTC w celu ich przetłumaczenia dla innego węzła końcowego. Podejście takie jest zdecydowanie bardziej oszczędne czasowo i zużywa mniej zasobów sprzętowych węzła

centralnego dzięki brakowi konieczności wykonywania operacji generowania klucza, jednakże jest mniej bezpieczne z powodu braku pewności co do wykorzystanego źródła entropii i procedur generowania klucza.

Dystrybucja klucza wykorzystująca rozwiązanie KTC przeprowadzana jest w 3 krokach, które opisane zostały na rysunku 2:

1) **$E(K_{N1}, Ks)$ –** wysłanie przez węzeł N1 wygenerowanego klucza Ks z żądaniem przetłumaczenia go dla klucza N2; żądanie zaszyfrowane kluczem głównym węzła N1.
2) **$E(K_{N2}, K_S)$ –** przesłanie do węzła N2 klucza Ks zaszyfrowanego kluczem głównym N2.
3) Testowanie przez węzły N1 i N2 klucza Ks.



Rys. 2. Schemat dystrybucji klucza metodą KTC

3. Zapewnienie wydajnej transmisji danych

Jedną z cech charakterystycznych urządzeń sieci IoT jest wymaganie dotyczące niskiego zużycia energii. Urządzenia IoT rzadko wykorzystują interfejsy umożliwiające komunikację w sieci Internet. W zdecydowanej większości wymiana danych odbywa się z wykorzystaniem Bluetooth, a coraz częściej również z użyciem innych specjalnych interfejsów komunikacyjnych. Obecnie istnieje wiele protokołów i interfejsów przeznaczonych do wymiany danych w sieciach IoT w różnych środowiskach. Każde z tych rozwiązań przeznaczone jest do innych celów i różnią się możliwościami, jednakże większość z nich ma wspólne cechy odróżniające je od protokołów i interfejsów używanych w sieci Internet:

- niskie zużycie energii, co ułatwia projektowanie urządzeń działających przez dłuższy czas na baterii,

- duży zasięg transmisji danych bezprzewodowych, sięgający kilkudziesięciu kilometrów,
- niska przepustowość łącza, rzadko przekraczająca 250 kb/s.

Ważną kwestią jest również fakt, że korzystanie z interfejsów i protokołów komunikacyjnych dla sieci IoT wiąże się z niską prędkością transmisji danych. W kontekście bezpieczeństwa sieci IoT jest to duży problem, ponieważ obsługa mechanizmów zapewniających poufność i integralność transmitowanych danych oraz zaufanie pomiędzy węzłami sieci wiąże się z dużym narzutem przesyłanych danych. W takich sytuacjach często można dostrzec duży spadek wydajności sieci.

3.1. Interfejsy komunikacyjne IoT

W trakcie projektowania sieci IoT najważniejszym czynnikiem wpływającym na wydajność transmisji danych jest wykorzystywany interfejs komunikacyjny. Ze względu na charakter i przeznaczenie takich sieci, często stosuje się w nich technologię LPWAN (ang. *Low Power Wide Area Network*). Technologia ta umożliwia urządzeniom IoT komunikację z wykorzystaniem bezprzewodowego medium transmisji, która charakteryzuje się niskim zużyciem energii oraz transmisją danych na duże odległości.

3.1.1. LoRa

Interfejs LoRa (ang *Long-Range*) jest przeznaczony do radiowej transmisji danych pomiędzy urządzeniami IoT o niskim poborze mocy. Charakteryzuje go niewielka prędkość transmisji danych sięgająca maksymalnie 50 kb/s. Zasięg sieci LoRa na otwartym terenie może sięgać do 28 km [3]. Do komunikacji wykorzystywane są następujące nielicencjonowane pasma częstotliwości radiowych:

- 169 MHz;
- 433 MHz;
- 868 MHz (Europa);
- 915 MHz (Ameryka Północna).

3.1.2. Sigfox

Jednym z najczęściej używanych interfejsów bezprzewodowych do transmisji danych w sieciach IoT jest Sigfox. Technologia Sigfox została zaprojektowana w taki sposób, aby zużycie energii urządzeń było minimalne i umożliwiało pracę na baterii do kilku lat. Dodatkowo interfejs ten umożliwia

transmisję danych na odległość sięgającą 50 km w obszarach wiejskich oraz do 10 km w obszarach zabudowanych [4]. Technologia Sigfox działa w jednym z dwóch pasm częstotliwości radiowej:

- 868 MHz;
- 902 MHz.

Urządzenia korzystające z Sigfox są zdolne transmitować dane z prędkością sięgającą około 0,1 kb/s. Tak niewielka przepustowość oznacza, że nadają się one jedynie do transmitowania takich danych, jak pomiary z różnego rodzaju czujników, natomiast nie znajdą zastosowania przy wymianie większej ilości danych. Dużą zaletą jest zasięg transmisji. Dzięki temu urządzenia wykorzystujące technologię Sigfox są idealne do monitorowania pracy i wykonywania pomiarów w sieciach przemysłowych, natomiast nie znajdą zastosowania przy wymianie bardziej złożonych danych.

3.1.3. XBee

Interfejs XBee jest interfejsem zapewniającym bezprzewodową transmisję danych pomiędzy urządzeniami IoT w paśmie 2,4 GHz. Moduł komunikacyjny XBee możne być podłączony do głównego urządzenia jednym z dwóch interfejsów, co bezpośrednio wpływa na prędkość transmisji danych:

- UART – umożliwia transmisję danych z prędkością 250 kb/s;
- SPI – umożliwia transmisję danych z prędkością 5 Mb/s.

W porównaniu do pozostałych interfejsów dla urządzeń IoT, prędkość transmisji danych zapewniona przez XBee jest najwyższa, jednakże maksymalny zasięg wynosi około 3 km [5]. Oznacza to, że jest to interfejs przeznaczony dla systemów wymagających szybkiej transmisji danych na stosunkowo niewielką odległość.

Tab. 1. Zestawienie interfejsów LPWAN

	LoRa	Sigfox	XBee
Zasięg transmisji (km)	28	Tereny wiejskie: 50 Tereny zabudowane: 10	3
Pasmo częstotliwości (MHz)	169 433 868 915	868 902	2400
Prędkość transmisji (kb/s)	50	0,1	250 (UART) 5000 (SPI)

3.2. Protokoły komunikacyjne IoT

Obecnie istnieje wiele protokołów komunikacyjnych przeznaczonych dla systemów IoT. Często ze względu na środowisko pracy nie ma możliwości, aby podłączyć urządzenia bezpośrednio do sieci Internet. Urządzenia komunikują się wówczas między sobą z wykorzystaniem specjalnych interfejsów komunikacyjnych (rozdział 3.1), a dodatkowo istnieje możliwość komunikowania się tych urządzeń z sieciami zewnętrznymi poprzez tzw. Bramki (ang. *Gateway*). Bramki stosowane są w przypadku każdego z protokołów IoT, a do jednej z ich funkcji należy translacja ramek na postać akceptowalną w sieci Internet, jeżeli protokół IoT nie obsługuje stosu protokołów TCP/IP.

3.2.1. Protokół LoRaWAN

LoRaWAN jest bezprzewodowym protokołem komunikacyjnym przeznaczonym do transmisji danych w rozległych sieciach IoT o niskiej przepustowości oraz dla urządzeń z ograniczonymi zasobami energetycznymi [6]. Protokół ten został zaprojektowany specjalnie dla interfejsu LoRa (punkt 3.1.1).

LoRaWAN ma na celu zapewnienie komunikacji urządzeniom, które według założenia powinny działać przez długi czas na jednej baterii. Sieci korzystające z tego protokołu są sieciami peer-2-peer, a kontakt z siecią Internet możliwy jest poprzez bramki. Nagłówek wykorzystywany przez LoRaWAN ma 13 bajtów.

Cechą charakterystyczną protokołu LoRaWAN jest brak komunikacji bezpośredniej pomiędzy urządzeniami końcowymi. Każda wymiana danych musi odbyć się poprzez serwer. Dodatkowo protokół LoRaWAN ma zaimplementowane szyfrowanie danych z wykorzystaniem 128-bitowych kluczy AES.

3.2.2. Protokół AMQP

Protokół AMQP (ang. *Advanced Message Queuing Protocol*) to protokół przeznaczony do wymiany danych pomiędzy aplikacjami w sposób asynchroniczny, a jego działanie bazuje na kolejkach. Protokół wyróżnia trzy elementy biorące udział w komunikacji:

- Exchange – pośrednik w wymianie wiadomości, obsługuje kolejki i podejmuje decyzje, w których kolejkach umieszczać wiadomości;
- Publisher – tworzy i wysyła wiadomości do Exchange;
- Consumer – klient, który pobiera wiadomości z kolejki.

AMQP jest protokołem asynchronicznym, co oznacza że odczyt wiadomości może nastąpić w dowolnym momencie i nie jest konieczne

nawiązywanie bezpośredniego połączenia pomiędzy dwoma węzłami końcowymi. Publisher w trakcie wysyłania wiadomości ustawia parametry wiadomości, które mogą być wykorzystane przez Exchange do ich obsługi. Nagłówek w protokole AMQP ma 8 bajtów.

Exchange na podstawie atrybutów wiadomości oraz wcześniej definiowanych zasad zwanych wiązaniami (ang. *bindings*) umieszcza wiadomości w odpowiednich kolejkach. Następnie wiadomości są dostarczane do Consumerów, którzy zasubskrybowali daną kolejkę bądź na ich żądanie. Exchange po otrzymaniu potwierdzenia odebrania wiadomości usuwa ją z kolejki, natomiast w przypadku niepowodzenia wiadomość może być zwrócona do Publishera, usunięta bądź przeniesiona do „*dead letter queue*" jeśli jest zainstalowane specjalne rozszerzenie. O podjętej czynności decyduje Publisher w trakcie ustawiania parametrów wiadomości.

Protokół AMQP cechuje wysoka wydajność, dzięki czemu jest w stanie obsłużyć dużą liczbę wiadomości w krótkim czasie. Dodatkowo zapewnione jest w nim bezpieczeństwo poprzez obsługę funkcji uwierzytelniania i szyfrowania. Ponadto protokół ten jest odporny na awarie. Negatywną cechą protokołu AMQP jest wymaganie dotyczące zasobów sprzętowych, w szczególności dla węzła Exchange, co w przypadku węzłów IoT nie zawsze jest możliwe.

3.2.3. Protokół CoAP

CoAP (ang. *Constrained Application Protocol*) jest protokołem działającym podobnie do modelu klient/serwer protokołu HTTP, jednakże węzły pełnią funkcję zarówno klienta, jak i serwera [7]. Każdy węzeł, który chce się skomunikować z innym, pełni funkcję klienta i wysyła zapytanie do drugiego węzła, który jako serwer udziela odpowiedzi. Zapytania wysyłane do serwera są podobne do zapytań HTTP. Wykorzystywane są metody (GET, POST, PUT, DELETE),
a odwoływanie się do zasobów odbywa się z wykorzystaniem URI (ang. *Uniform Resource Identifier*).

CoAP został zaprojektowany specjalnie dla urządzeń o ograniczonych zasobach, w związku z czym wiadomości są skondensowane, a nagłówki mają 4 bajty. Dzięki temu wymagana jest mniejsza przepustowość sieci oraz oszczędzane są zasoby urządzeń. Co więcej, CoAP ma zaimplementowane mechanizmy bezpieczeństwa, takie jak obsługa protokołu DTLS (ang. *Datagram Transport-Layer Security*), który zapewnia szyfrowanie i uwierzytelnianie danych.

3.2.4. Protokół MQTT

Protokół MQTT (ang. *Message Queuing Telemetry Transport*) jest lekkim protokołem komunikacyjnym przeznaczonym dla sieci IoT. MQTT zostało zaprojektowane z myślą o urządzeniach IoT dysponujących niewielkimi zasobami pamięciowymi i obliczeniowymi. Dzięki niskim wymaganiom protokołu w kwestii dotyczącej zasobów, MQTT umożliwia wymianę danych pomiędzy urządzeniami bez większego wpływu na ich wydajność.

Protokół MQTT działa na zasadzie publikacji i subskrypcji. Występują w nim trzy typy urządzeń: MQTT Broker, który jest serwerem pośredniczącym w wymianie danych, MQTT Publisher, który wysyła dane do brokera oraz MQTT Subscriber, który pobiera interesujące go dane do Brokera. Broker przechowuje listę tematów, odnośnie do których może przyjmować i wysyłać wiadomości. Publisher może opublikować swoje dane na dany temat, który jest umieszczany w nagłówku wiadomości, natomiast subscriber może te dane pobrać, jeśli jest zainteresowany tematem. Dzięki temu minimalizowana jest liczba przesyłanych wiadomości.

Protokół MQTT cechuje duża skalowalność, dzięki której możliwe jest przesyłanie danych do wielu odbiorców jednocześnie. Dodatkowo możliwe jest łatwe dodawanie nowych urządzeń do sieci bez konieczności zmiany jej wcześniejszej konfiguracji. Ponadto urządzenia korzystające z MQTT dzięki minimalnej liczbie przesyłanych wiadomości ograniczają w ten sposób zużycie energii do minimum.

Do największych wad protokołu MQTT należy brak mechanizmów bezpieczeństwa. MQTT nie zapewnia poufności ani integralności danych, w związku z czym w celu zapewnienia bezpieczeństwa danych konieczna jest implementacja niezależnych protokołów, takich jak TLS.

3.2.5. Protokół ZigBee

Protokół ZigBee został zaprojektowany dla urządzeń IoT o niskim poborze mocy i pracujących w sieciach, w których komunikacja odbywa się z wykorzystaniem interfejsu XBee. ZigBee wymagania istnienia dwóch typów węzłów: koordynatora, czyli węzła sieciowego odpowiedzialnego za inicjację połączeń, zarządzanie siecią i przesyłanie wiadomości pomiędzy urządzeniami oraz urządzenia końcowego, pełniącego różne funkcje.

ZigBee ma warstwową budowę, gdzie każda z warstw pełni inne funkcje i jest niezależna od innych, a jednocześnie ściśle współpracuje z sąsiadującymi warstwami. Można wyróżnić cztery warstwy: warstwę fizyczną (PHY), warstwę MAC (ang. *Medium Access Control*), warstwę sieciową (NWK) oraz warstwę aplikacji (APL). Protokół ZigBee, dzięki rozbudowanej warstwie NWK, jest w stanie obsłużyć topologię gwiazdy, drzewa oraz mesh.

Protokół ZigBee ma zaimplementowane funkcje kryptograficzne podnoszące poziom bezpieczeństwa sieci. Transmitowane wiadomości zabezpieczane są szyfrowaniem AES z kluczem 128-bitowym. Dodatkowo każde z urządzeń musi przejść autoryzację, aby dołączyć do sieci i uczestniczyć w wymianie danych, co zapewnia budowę zaufanego środowiska już na wczesnym etapie funkcjonowania sieci.

Podsumowanie

We współczesnych sieciach IoT stosuje się wiele różnych rozwiązań, a każde z nich charakteryzują inne właściwości. Wybór odpowiednich technologii i mechanizmów uwarunkowany jest funkcjami, jakie dana sieć ma pełnić.

W kontekście bezpieczeństwa ważne jest, aby sieć już od początku funkcjonowania zapewniała zaufanie pomiędzy węzłami sieci. Z tego powodu konieczne jest wdrożenie wydajnych i skutecznych mechanizmów uwierzytelniania. W zależności od wymagań stawianych sieciom przez administratorów, do których należy między innymi stopnień zautomatyzowania węzłów, możliwy jest wybór różnych mechanizmów oferujących różne funkcje. Mechanizmy takie mogą obejmować uwierzytelnianie z wykorzystaniem kontekstów, haseł oraz kryptografii asymetrycznej.

Kolejnym ważnym aspektem jest zapewnienie poufności transmitowanych danych. W tym celu zazwyczaj korzysta się z kryptografii symetrycznej ze względu na wymaganą mniejszą moc obliczeniową i mniejsze zasoby pamięciowe niż w przypadku kryptografii asymetrycznej. Stosowanie kryptografii symetrycznej wiąże się jednak z problemem bezpiecznego rozpowszechnienia klucza symetrycznego, szczególnie gdy konieczne jest zapewnienie jednego klucza dla wielu węzłów. Z tego powodu konieczne jest wdrożenie odpowiednio zaprojektowanego skalowalnego systemu dystrybucji kluczy symetrycznych.

Istnieją dwa podejścia dotyczące dystrybucji kluczy symetrycznych: KDC oraz KTC. Każde z tych rozwiązań skupia się na innych aspektach. KDC jest rozwiązaniem scentralizowanym, gdzie najważniejsze operacje kryptograficzne wykonywane są na węźle centralnym. KTC jest podejściem zdecentralizowanym, gdzie każdy z kluczy jest generowany na węzłach końcowych, a węzeł centralny odpowiada jedynie za przekazanie gotowego klucza do innych węzłów, a brak konieczności generowania klucza odciąża węzeł centralny i umożliwia szybsze obsługiwanie kolejnych żądań.

Ostatnią kwestią poruszaną w artykule jest transmisja danych pomiędzy węzłami IoT. Ze względu na ograniczony dostęp do zasobów energetycznych urządzenia wykorzystują specjalne interfejsy komunikacyjne i protokoły wymiany danych. Każde z tych rozwiązań cechuje inny zasięg oraz prędkość transmisji danych, w związku z czym wybór odpowiednich rozwiązań

podyktowany jest środowiskiem, w jakim działa sieć oraz zadaniami, jakie mają w ramach takiej sieci być realizowane.

Literatura

[1] Needham R. M., Schroeder M. D., *Using encryption for authentication in large networks of computers*, Communications of the ACM, vol. 21, no. 12, 1978, pp. 993-999.

[2] Barker E. and Baker W., *Recommendation for Key Management: Part 2 – Best Practices for Key Management Organizations*, NIST Special Publication 800-57 Revision 1. May, 2019.

[3] Jovalekic N., Drndarevic V., Pietrosemoli E., Darby I., Zennaro M., *Experimental Study of LoRa Transmissionover Seawater*, Sensors, 18 (9), 2853, 2018.

[4] Augustin A., Yi J., Clausen T., Townsley W. M., *A Study of LoRa: Long Range & Low Power Networks for the Internet of Things*, Sensors, 16 (9), 1466, 2016.

[5] Maciaszczyk R., *Transmisja danych z pojazdów bezzałogowych z wykorzystaniem modułów XBee*, Pomiary Automatyka Kontrola, R. 58, nr 2, 2012, s. 656-658.

[6] Farrell S., [RFC 8376], *Low-Power Wide Area Network (LPWAN) Overview*, 2018.

[7] Shelby A., Hartke K., Bormann C., [RFC 7252], *The Constrained Application Protocol (CoAP)*, 2014.

Secure and efficient data exchange in Internet of Things networks – technology review

ABSTRACT: IoT devices are often accompanied by numerous hardware constraints, such as limited computing power, limited memory resources, and low network bandwidth. All these components have a significant impact on the difficulties in ensuring the security of IoT networks. For this reason, various solutions designed specifically for IoT devices have been developed to increase the level of security of such devices. The paper reviews the most popular solutions used in IoT networks that minimize problems caused by numerous limitations.

KEYWORDS: IoT, IoT network security, key distribution, lightweight data exchange protocols, IoT communication interfaces

Praca wpłynęła do redakcji: 15.05.2023 r.