Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer e-ISSN: 2548-964X
Vol. 2, No. 10, Oktober 2018, hlm. 4150-4155 http://j-ptiik.ub.ac.id

Analisis Mekanisme End-To-End Security Pada Komunikasi Antara Node Sensor Dengan IoT Middleware

Ahmad Lutfi Bayu Aji[1], Eko Sakti Pramukantoro[2], Reza Andria Siregar[3]

Program Studi Teknik Informatika, Fakultas Ilmu Komputer, Universitas Brawijaya
Email: [1]bayuaji732@gmail.com, [2]ekosakti@ub.ac.id, [3]reza@ub.ac.id

Abstrak

Internet of things (IoT) adalah salah satu teknologi masa depan yang mengalami perkembangan pesat. IoT bisa menggunakan data yang telah dikumpulkan tanpa bantuan manusia, yang kemudian saling berhubungan satu sama lain dengan internet. Pada penelitian sebelumnya telah dikembangkan sebuah *middleware* dengan pendekatan *event-driven* yang mampu mendukung interoperabilitas berbagai macam perangkat atau sensor. *Middleware* terdiri dari sensor nodemcu ESP8266 dan Raspberry Pi sebagai *broker/server*. Sensor berkomunikasi menggunakan protokol MQTT dan CoAP dengan *middleware*. Setiap jaringan komputer pasti memiliki kerentanan. Salah satu serangan pada komunikasi jaringan adalah *Man-In-The-Middle* (MITM). Pada skema *middleware* tersebut fitur keamanan belum diterapkan baik pada transmisi data ataupun validasi data sehingga dibutuhkan metode untuk mengatasi celah keamanan tersebut atau bisa disebut *end-to-end security*. Pada penelitian ini akan ditambahkan mekanisme TLS dan crypto AES-CBC untuk melindungi data yang dikirim dari sensor nodemcu ke *middleware*. Dan mendapatkan rekomendasi mekanisme keamanan yang sesuai dengan spesifikasi sensor nodemcu ESP8266 yang minim. Hasilnya, TLS gagal diterapkan sedangkan crypto AES-CBC 128 bits berhasil dan tidak mempengaruhi QoS pada pengiriman data. Sehingga direkomendasikan untuk diterapkan pada *middleware* sebagai *end-to-end security*.

Kata kunci: *Internet of Things*, *middleware*, ESP8266, TLS, AES-CBC, QoS

Abstract

Internet of things (IoT) is one of the technology of the future that develops rapidly. IoT can use data that has been collected without human help, which then interconnected with each other with the internet. In the previous research, a middleware with an event-driven approach has been developed that can support the interoperability of various devices or sensors. Middleware consists of sensor nodemcu ESP8266 and Raspberry Pi as a broker / server. Sensors communicate using MQTT and CoAP protocols with middleware. Every computer network must have vulnerabilities. One of the attacks on network communication is Man In The Middle (MITM). In the middleware scheme, the security features have not been applied either to data transmission or data validation, so there is a need to overcome the security hole or end-to-end security. In this research we will add TLS and AES-CBC crypto mechanisms to protect data sent from nodemcu sensor to middleware. And get recommendations of security mechanisms that match the specifications of the Nodemcu ESP8266 minimal sensor. As a result, TLS fails to be applied while AES-CBC crypto succeeds and does not affect QoS on data transmission. It is recommended to apply to middleware as end-to-end security.

Keywords: *Internet of Things, middleware, ESP8266, TLS, AES-CBC, QoS*

1. PENDAHULUAN

Pada penelitian sebelumnya telah dikembangkan sebuah lingkungan *middleware*. *Middleware* dengan pendekatan *event-drivent* tersebut mendukung interoperabilitas untuk perangkat dan sensor yang bermacam-macam (Anwari, 2017). *Middleware* tersebut menyediakan *gateway* berupa protokol MQTT dan CoAP untuk berkomunikasi dengan sensor dan menggunakan websocket untuk berkomunikasi dengan aplikasi lain. MQTT dan CoAP adalah sebuah protokol standar yang digunakan pada IoT. Sensor yang digunakan

adalah nodemcu ESP8266 dan Raspberry pi sebagai *broker/server* (Anwari, 2017). Penelitian lain, telah dikembangkan *data storage framework* untuk *middleware* yang sama (Pramukantoro, 2017). Pada komunikasi jaringan komputer pasti ada kerentanan. Salah satu serangan pada jaringan adalah *Man-in-the-Middle* (MITM). Pada skema *middleware* tersebut fitur keamanan belum diterapkan baik pada transmisi data ataupun validasi data sehingga dibutuhkan metode untuk mengatasi celah keamanan tersebut atau bisa disebut *end-to-end security*.

End-to-end (E2E) security adalah sebuah protokol dan mekanisme yang fokus untuk melindungi titik akhir dari koneksi. *Endpoints* atau titik akhir koneksi ini biasa disebut sebagai klien atau server. TLS dan AES-128 CBC adalah dua metode utama yang digunakan pada *E2E security* (www.cisco.com, 2009).

TLS adalah Protokol yang memungkinkan klien/server aplikasi untuk berkomunikasi dengan cara yang dirancang untuk mencegah penyadapan, gangguan atau pemalsuan pesan. AES adalah sebuah blok cipher yang mana mengenkripsi satu blok data pada satu waktu (Vaidehi, 2014).

Berdasarkan uraian di atas, akan diterapkan mekanisme TLS dan crypto AES-CBC sebagai E2E pada IoT middleware. Melihat minimnya spesifikasi sensor nodemcu ESP8266 diharapkan ada rekomendasi dari kedua mekanisme ini yang paling sesuai, aman dan efisien serta tidak mengurangi QoS pengiriman data dari sistem sebelumnya.

2. LANDASAN KEPUSTAKAAN

2.1. Kajian Pustaka

Terdapat penelitian sebelumnya yang terkait dengan penelitian ini, berjudul *Taxonomy of Man-in-the-Middle Attacks on HTTPS* (Shaun, dkk, 2016). Pada penelitian tersebut dilakukan klasifikasi dan pencegahan terhadap serangan MITM pada HTTPS.

Penelitian lain dengan judul *End-to-End IoT Security Middleware for Cloud-Fog Communication* (Mukherjee, dkk, 2017) dilakukan desain dan implementasi pada *middleware* dengan *E2E security*.

Penelitian lain dengan judul Topic *based IoT data storage framework for heterogeneous sensor data* (Pramukantoro, 2017). Penelitian tersebut membuat *framework* untuk penyimpanan data pada *middleware* yang sama dengan penelitian ini, menggunakan MongoDB dan GridFS.

2.2 *Middleware*

Middleware adalah sebagai antarmuka antara lapisan perangkat keras dan lapisan aplikasi, yang bertanggung jawab untuk berinteraksi dengan perangkat dan manajemen informasi (Abdur, 2016). Middleware yang digunakan pada penelitian ini adalah middleware yang dikembangkan oleh Husnul Anwari.

2.3 TLS

Transport Layer Security (TLS) dimaksudkan untuk memberikan keamanan saluran antara dua entitas berkomunikasi melalui internet. Entitas menggunakan sertifikat X.509 untuk pertukaran kunci publik. Kunci publik lebih baik digunakan untuk otentikasi. Kemudian digunakan untuk menukar kunci sesi karena seperti yang kita ketahui publik kriptografi kunci tidak dapat digunakan secara langsung untuk enkripsi dan dekripsi (karena perhitungan overhead) jadi ada kebutuhan kunci simetris atau kunci sesi di TLS. Jadi kunci sesi adalah digunakan untuk enkripsi data yang mengalir antara berkomunikasi entitas. Ini menyediakan kerahasiaan data dan juga menggunakan MAC (kode otentikasi pesan) untuk otentikasi pesan (Ranjan, 2014).

2.4 AES

AES adalah algoritme kunci simetris, keduanya pengirim dan penerima menggunakan satu kunci untuk enkripsi dan
dekripsi AES mendefinisikan panjang blok data menjadi 128 bit, dan Panjang kunci sampai 128.192, atau 256 bit. Ini adalah sebuah iteratif Algoritma dan setiap iterasi disebut round. Jumlah seluruhnya jumlah putaran, Nr, adalah 10, 12, atau 14 bila panjang kunci adalah 128, 192, atau 256 bit, masing-masing (Bhajaj, 2016).

2.5 MITM

Dalam serangan MITM, skenario umum melibatkan: dua titik akhir (korban), dan pihak ketiga (penyerang). Penyerang memiliki akses pada saluran komunikasi antara dua titik akhir, dan bisa memanipulasi pesan mereka (Conti, 2016).

3. METODOLOGI PENELITAN

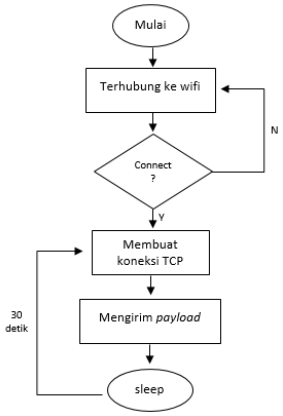
Metodologi penelitian dengan identifikasi

masalah, studi literatur kemudian persiapan testbed dan perancangan lingkungan. Pengujian dan pengambilan data dilakukan untuk mengetahui apakah sistem sudah berjalan sesuai dengan perancangan lingkungan. Kemudian hasil dari pengujian dilakukan analisis pada pembahasan untuk kemudian diambil kesimpulan.

3.1. Perancangan Lingkungan

Perancangan lingkungan dilakukan untuk mengetahui bagaimana sistem akan berjalan sehingga siap dilakukan pengujian dan pengambilan data.

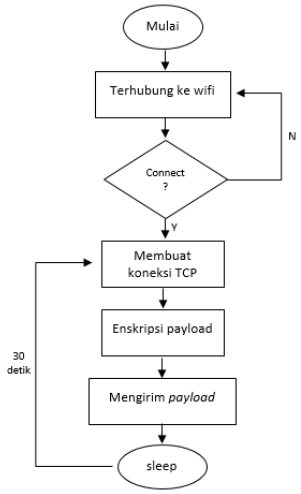
Pengiriman Data Protokol MQTT Dan CoAP Tanpa Mekanisme Keamanan



Gambar 1. Diagram alir tanpa keamanan

Script lua yang berisi pengiriman data dengan protokol MQTT dan CoAP di-*upload* pada sensor nodeMcu ESP8266. Data yang dikirim adalah data *dummy* dari *temperature* dan *humidity*. Data yang dikirim adalah dalam bentuk JSON. Pertama sensor nodemcu akan berkoneksi dengan wifi dari *middleware*. Kemudian membuat koneksi TCP untuk berhubungan dengan protokol MQTT dan CoAP dari dari *middleware*. Kemudian data *dummy* akan di-*encode* ke dalam JSON sebagai *payload* dan di-*Publish* oleh sensor. Setelah pengiriman, sensor akan *idle* selama 30 detik sebagai waktu istirahat untuk pengiriman data selanjutnya.

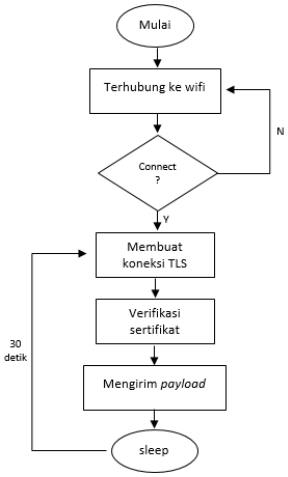
Pengiriman Data Protokol MQTT Dan CoAP Dengan AES-CBC



Gambar 2. Diagram alir dengan crypto

Script lua yang berisi pengiriman data dengan protokol MQTT dan CoAP di-*upload* pada sensor nodeMcu ESP8266. Data yang dikirim adalah data *dummy* dari *temperature* dan *humidity*. Data yang dikirim adalah dalam bentuk JSON. Pertama sensor nodemcu akan berkoneksi dengan wifi dari *middleware*. Kemudian membuat koneksi TCP untuk berhubungan dengan protokol MQTT dan CoAP dari dari *middleware*. Kemudian data *dummy* akan di-*encode* ke dalam JSON sebagai *payload* dan dienkripsi dengan algoritma AES-CBC lalu di-*Publish* oleh sensor. Setelah pengiriman, sensor akan *idle* selama 30 detik sebagai waktu istirahat untuk pengiriman data selanjutnya.

Pengiriman Data Protokol MQTT Dan CoAP Dengan TLS



Gambar 3. Diagram alir dengan TLS

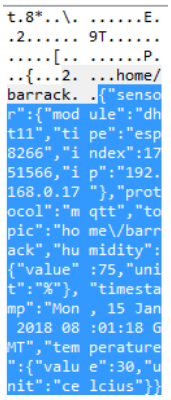
Script lua yang berisi pengiriman data dengan protokol MQTT di-*upload* pada sensor nodeMcu ESP8266. Data yang dikirim adalah

data *dummy* dari *temperature* dan *humidity*. Data yang dikirim adalah dalam bentuk JSON. Pertama sensor nodemcu akan berkoneksi dengan wifi dari *middleware*. Kemudian membuat koneksi TLS untuk berhubungan dengan protokol MQTT dari dari *middleware* dan melakukan verifikasi sertifikat. Kemudian data dummy akan di-*encode* ke dalam JSON sebagai *payload* di-*Publish* oleh sensor dengan koneksi yang aman. Setelah pengiriman, sensor akan *idle* selama 30 detik sebagai waktu istirahat untuk pengiriman data selanjutnya.

4. Pengujian dan Pembahasan

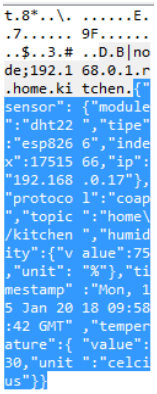
Berdasarkan pada diagram alir perancangan lingkungan pada bab sebelumnya. Berikut adalah skenario-skenario pengujian pengiriman data tersebut.

Skenario Pengiriman Data Protokol MQTT Dan CoAP Tanpa Mekanisme Keamanan



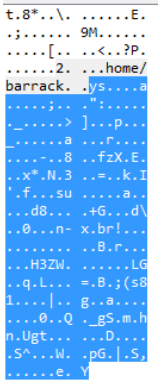
Gambar 4. Pengiriman data protokol MQTT tanpa keamanan

Pengiriman dilakukan dengan mengirimkan data dari sensor nodemcu selama 10 menit ke *middleware* dengan jeda 30 detik dan lima kali pengujian. Dan menjalankan program tcpdump untuk meng-*capture* transmisi data yang terjadi. Hasil dari tcpdump diolah di wireshark. Output yang didapat adalah data berhasil dikirim dari nodemcu ke *middleware* dengan protokol MQTT dan CoAP. Namun data mudah dibaca oleh penyerang sehingga bisa disalahgunakan dan merugikan korban.



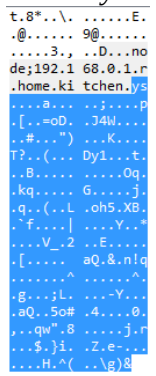
Gambar 5. Pengiriman data protokol MQTT tanpa keamanan

Skenario Pengiriman Data Protokol MQTT Dan CoAP Dengan Crypto



Gambar 6. Pengiriman data protokol MQTT dengan crypto

Pengiriman dilakukan dengan mengirimkan data dari sensor nodeMcu selama 10 menit ke *middleware* dengan jeda 30 detik dan lima kali pengujian. Dan menjalankan program tcpdump untuk meng-*capture* transmisi data yang terjadi. Hasil dari program tcdump kemudian diolah dengan wireshark. Output yang didapat adalah data berhasil dikirim dari nodeMcu ke *middleware* dengan protokol MQTT dan CoAP telah dienkripsi. Data sulit dibaca sehingga telah aman dari segi *confidentiality*.



Gambar 7. Pengiriman data protokol MQTT dengan crypto

Skenario Pengiriman Data Protokol

MQTT Dan CoAP Dengan TLS

Pengiriman dilakukan dengan mengirimkan data dari sensor nodeMcu selama 10 menit ke *middleware* dengan jeda 30 detik dan lima kali pengujian. Dan menjalankan program tcpdump untuk meng-*capture* transmisi data yang terjadi. Hasil dari program tcdump kemudian diolah dengan wireshark. *Output* yang didapat adalah data tidak bisa dikirim dari sensor nodemcu ke *middleware* dengan protokol MQTT dan CoAP tidak terjadi verifikasi sertifikat. Implementasi TLS gagal dilakukan karena *handshake failed*.

Tabel 1. Perbandingan Hasil pengujian

Mekanisme	Delay(detik)	Jitter (detik)
MQTT	0.018039	0.000269
CoAP	0.019705	0.000664
MQTT crypto	0.017027	0.0003894
COAP crypto	0.0233	0.000742

Pada tabel 1 menunjukkan bahwa crypto AES tidak mempengaruhi QoS pada pengiriman data. Karena data dienkripsi terlebih dahulu sebelum dikirim.

5. KESIMPULAN

Berdasarkan analisis yang telah dilakukan sebelumnya, kesimpulan yang didapat sebagai berikut:

1. Penelitian dilakukan pada lingkungan middleware yang merupakan tiruan dari middleware yang dikembangkan sebelumnya oleh Husnul Anwari. TLS dan *crypto* AES-CBC 128 bits ditambahkan pada komunikasi protokol MQTT dan COAP antara node sensor dengan *middleware* sebagai end-to-end security. TLS dan crypto AES-CBC 128 bits masing-masing ditambahkan pada kedua protokol MQTT dan CoAP dengan modul dari sensor nodemcu ESP8266.
2. Berdasarkan hasil dari pengujian, implementasi modul *crypto* berhasil mengirimkan data yang telah dienkripsi dengan algoritma AES-CBC dari node sensor ke *middleware* dengan kedua protokol MQTT dan CoAP. Sedangkan TLS belum bisa diimplementasikan karena masalah dari modul TLS nodemcu. Koneksi klien terus menutup dan memulai koneksi baru setelah server mengirim balasan server *hello*. Sehingga verifikasi sertifikat yang dibutuhkan server untuk melakukan koneksi TLS tidak bisa dilakukan.
3. Berdasarkan hasil pengujian, mekanisme *crypto* AES-CBC 128 bits pada nodeMcu tidak memberikan perbedaan pada QoS dengan pengiriman data tanpa mekanisme keamanan. Karena data dienkripsi terlebih dahulu sebelum data dikirim. Nilai delay protokol MQTT tanpa dan dengan *crypto* adalah 0.018039 detik dan 0.0170278 detik . Nilai delay protokol CoAP tanpa dan dengan *crypto* adalah 0.019705 detik dan 0.02333 detik. Dan nilai *jitter* protokol MQTT tanpa dan dengan *crypto* adalah 0.000269 detik dan 0.0003894 detik . Nilai *jitter* protokol CoAP tanpa dan dengan *crypto* adalah 0.000664 dan 0.000742. Protokol CoAP dengan *crypto* lebih baik dibandingkan MQTT dengan *crypto* karena memiliki rata-rata *packet loss* yang lebih sedikit yaitu 4% banding 24%.

Saran pada penelitian mendatang adalah bisa menerapkan mekanisme E2E *security* yang berbeda dan menggunakan parameter dan lingkungan sistem yang berbeda.

6. DAFTAR PUSTAKA

Abdur Razzaque M, Milojevic-jervic M, Palade A, Clake S., 2016. *Middleware for Internet of Things: A Survey*.

Anwari H., 2017. *Pengembangan IoT Middleware Berbasis Eventbase Dengan Potokol Komunikasi CoAP, MQTT dan Websocket*. Malang. Universitas Brawijaya.

Bhajaj R.D, Gokhale Dr U.M., 2016. *AES ALGORITHM FOR ENCRYPTION*. Nagpur. G.H. Raisoni Institute of Engineering and Technology for Women.

Mukherjee B, Roshan, Prasad. 2017. *End-to-End IoT Security Middleware for Cloud-Fog Communication*. USA. University of Missouri-Columbia.

Pramukantoro E.S, W. Yahya, G. Arganata, A. Bhawiyuga and A. Basuki, "Topic based IoT data storage framework for heterogeneous sensor data" 2017 11th International Conference on Telecommunication Systems Services and Applications (TSSA), Lombok, Indonesia, 2017, pp. 1-4. doi: 10.1109/TSSA.2017.8272895

Ranjan A.K, Vijay Kumar, Muzzammil Hussain. 2014. *Security Analysis of TLS Authentication*. India. Central University of Rajasthan.

Vaidehi M, Dr. B.Justus R. 2014. *Design and Analysis of AES-CBC Mode for High*

Security Applications. Coimbatore. Karpagam University.