RANCANG BANGUN SISTIM KEAMANAN PERANGKAT IoT DENGAN METODE AUTENTIKASI MENGGUNAKAN JSON WEB TOKEN PADA PROTOKOL MQTT

Celvin Arya Mangkurat

Program Studi Teknik Multimedia dan Jaringan, Jurusan Teknik Informatika dan Komputer, Politeknik Negeri Jakarta

Email: celvinarya10@gmail.com

Abstrak

IoT merupakan suatu konsep yang bertujuan untuk menghubungkan perangkat-perangkat yang ada disekitar melalui internet sehingga dapat menciptakan sebuah lingkungan yang cerdas dengan melakukan proses pengumpulan dan pertukaran data antar *node*. Dalam mendukung proses pertukaran data pada perangkat IoT agar cepat dan ringan maka dibutuhkan sebuah protokol yang dapat bekerja dengan energi dan media penyimpanan yang kecil. Protokol MQTT adalah protokol pesan ringan berbasis *publish-subscribe* dengan ukuran paket data yang kecil dan juga konsumsi catu daya kecil. Oleh karena itu, protokol MQTT cocok untuk diterapkan pada perangkat IoT. Walaupun demikian, protokol tersebut masih rentan terhadap serangan siber. Berdasarkan survey yang dilakukan oleh Shodan, terdapat hampir 67.000 server MQTT yang beredar di internet dengan sebagian besar tidak memiliki autentikasi. Oleh karena itu, penelitian ini dilakukan untuk mengatasi permasalah tersebut dengan menerapkan metode autentikasi menggunakan JSON Web Token dan protokol TLS. Berdasarkan hasil pengujian yang dilakukan dapat diketahui bahwa sistem ini dapat melakukan autentikasi terhadap token valid, kedaluwarsa dan tanda tangan yang *invalid*. Selain itu, dapat diketahui NodeMCU ESP8266 membutuhkan waktu untuk memperoleh token paling sedikit yaitu 0,04 detik, paling banyak yaitu 0,991 detik dan rata-rata waktu yang dibutuhkan adalah 0,43028 detik. Kemudian, dengan menerapkan protokol keamanan TLS, sistem ini dapat menangkal serangan *man in the middle*.

Kata kunci: autentikasi, JSON web token, keamanan IoT, MQTT, serangan *man in the middle*

Abstract

IoT is a concept that aims to connect nearby devices via the internet so that it can create a smart environment by carrying out the process of collecting and exchanging data between nodes. In supporting the process of exchanging data on IoT devices to be fast and light, we need a protocol that can work with small energy and storage media. The MQTT protocol is a publish-subscribe-based lightweight messaging protocol with small data packet size and low power consumption. Therefore, the MQTT protocol is suitable for IoT devices. However, the protocol is still vulnerable to cyber attacks. Based on a survey conducted by Shodan, there are nearly 67,000 MQTT servers circulating on the internet, most of which have no authentication. Therefore, this research was conducted to overcome this problem by implementing an authentication method using JSON Web Token and the TLS protocol. Based on the results of the tests carried out, it can be seen that this system can authenticate valid tokens, expired tokens and token with invalid signature. In addition, it can be seen that the NodeMCU ESP8266 takes a minimum of 0.04 seconds to get a token, a maximum of 0.991 seconds and an average time of 0.43028 seconds. Then, by implementing the TLS security protocol, the system can ward off man-in-the-middle attacks.

***Keywords**: authentication, JSON web token, IoT security, MQTT, man in the middle attack*

1. PENDAHULUAN

Internet of Things (IoT) merupakan suatu konsep yang bertujuan untuk menghubungkan perangkat-perangkat yang ada disekitar melalui internet sehingga dapat menciptakan sebuah lingkungan yang cerdas dengan melakukan proses pengumpulan dan pertukaran data antar node. Perangkat-perangkat tersebut biasanya terhubung dengan mikrokontroler, sensor dan juga koneksi internet (Kashyap, Sharma & Gupta, 2018). Dalam mendukung proses pertukaran data pada perangkat IoT agar cepat dan ringan maka dibutuhkan sebuah protokol yang dapat bekerja dengan energi dan media penyimpanan yang kecil. Protokol *Message Queuing Telemetry Transport* (MQTT) adalah protokol pesan ringan (*lightweight*) berbasis *publish-subscribe* yang berjalan di atas protokol TCP/IP (Boyd dkk., 2014). Protokol ini mempunyai ukuran paket data *low overhead* kecil dan juga konsumsi catu daya kecil. MQTT bersifat terbuka, simpel dan didesain agar mudah untuk diimplementasikan. MQTT dapat menangani ribuan pengguna secara jarak jauh hanya dengan menggunakan satu *broker*. (Lampkin dkk., 2012). Karakteristik inilah yang membuat protokol MQTT ideal untuk digunakan pada perangkat IoT. Walaupun demikian, protokol tersebut masih rentan terhadap serangan siber (Singh dkk., 2015).

Penyebab utama masalah keamanan pada IoT adalah karena konfigurasi *default* yang tidak aman dan tidak adanya autentikasi. Setiap objek pada IoT harus dapat mengidentifikasi dan mengautentikasi objek lain agar dapat membangun proses pertukaran data yang aman (Mahmoud dkk., 2016). Selain itu, *wildcard* pada MQTT memungkinkan pengguna yang memiliki akses ke server dapat mengakses ke semua pesan yang mengalir melaluinya sehingga dibutuhkan sebuah konfigurasi pada MQTT *broker* yang benar. Berdasarkan survey yang dilakukan oleh shodan, terdapat hampir 67.000 server MQTT yang beredar di internet dengan sebagian besar tidak memiliki autentikasi. Secara konfigurasi *default*, protokol MQTT masih rentan terhadap serangan *man in the middle* yang dapat mencuri informasi penting seperti nama pengguna, kata sandi hingga isi pesan yang di-*publish* dan juga memungkinkan siapa saja untuk berlangganan topik yang disiarkan tanpa adanya autentikasi sehingga mereka dapat menerima seluruh pesan berdasarkan topik yang telah di-*subscribe* (Lundgren, 2016).

Berdasarkan permasalahan tersebut maka diterapkanlah protokol *Transport Layer Security* (TLS) yang dapat menangkal serangan *man in the middle* dan metode autentikasi yang dapat memvalidasi bahwa data yang dikirim berasal dari pengirim yang sah. JSON Web Token (JWT) merupakan *open standard* (RFC 7519) yang digunakan untuk mengirimkan data secara padat dan aman sebagai objek JSON (Shingala, 2019) (Jones, Bradley & Sakimura, 2015) (Ahmed & Mahmood, 2019). Kemampuan JWT untuk melakukan verifikasi dan tanda tangan secara digital dapat mencegah terjadinya pengiriman data dari pengguna yang tidak terotorisasi (Bhawiyuga, Data & Warda, 2018). JWT digunakan sebagai metode autentikasi sehingga server dapat melakukan verifikasi bahwa data yang dikirim berasal dari pengguna yang sah. Apabila terdeteksi bahwa Token JWT yang dikirim tidak valid maka server akan mengirimkan notifikasi melalui aplikasi Telegram. Telegram dipilih untuk digunakan sebagai media notifikasi pada sistem ini karena Telegram merupakan aplikasi layanan pesan berbasis *cloud* dan bersifat terbuka yang telah menyediakan enkripsi *end-to-end*, *self destruction messages*, dan infrastruktur *multi data center*. Fitur-fitur tersebut lah yang membuat proses komunikasi pada aplikasi Telegram menjadi aman (Panjaitan & Syafari, 2019) (Fahana, Umar & Ridho, 2017). Oleh karena itu, dianggap penting untuk mengangkat penelitian mengenai keamanan perangkat IoT dengan metode autentikasi menggunakan JSON Web Token pada protokol MQTT.

2. TINJAUAN PUSTAKA

2.1. *Internet of Things* (IoT)

IoT dapat didefinisikan sebagai kemampuan beberapa perangkat yang dapat saling berkomunikasi, terhubung dan bertukar data melalui jaringan internet. IoT merupakan sebuah teknologi komunikasi yang memungkinkan adanya pengendalian, komunikasi, kerjasama dengan berbagai perangkat–perangkat keras, berkomunikasi data melalui jaringan internet. Sehingga bisa disimpulkan bahwa IoT merupakan sebuah konsep menyambungkan atau menghubungkan sesuatu, yang tidak dioperasikan oleh manusia ke internet. Konsep dasar dari internet of things adalah dengan menggabungkan

objek, sensor, kontroler, dan internet yang bisa menyebarkan informasi kepada pengguna. Objek akan dideteksi oleh sensor yang akan diproses oleh kontroler dan dilanjutkan untuk mengirim data yang sudah diolah sehingga menjadi sebuah informasi yang berguna dan secara *real-time* kepada pengguna (Li, Xu dan Zhao, 2015).

2.2. *Message Queuing Telemetry Transport*

Protokol *Message Queuing Telemetry Transport* (MQTT) adalah protokol yang berjalan pada *layer* aplikasi dan dirancang untuk perangkat dengan sumber daya terbatas. MQTT berbasis pada topik dengan arsitektur *publish-subcribe*. MQTT menggunakan protokol TCP/IP sebagai standar untuk proses pertukaran data. Protokol ini memiliki ukuran paket data *low overhead* (maksimal 2 gigbyte) dengan konsumsi catu daya kecil. MQTT bersifat *open source*, didesain agar mudah diimplementasikan dan mampu menangani ribuan klien secara jarak jauh dengan hanya menggunakan satu server(Yudha Saputra dkk., 2017). Arsitektur *publish/subscribe* membutuhkan *broker* yang bertanggung jawab untuk mendistribusikan pesan kepada klien yang telah melakukan *subscribe* terhadap topik pesan (Locke, 2010).

2.3. *Transport Layer Security*

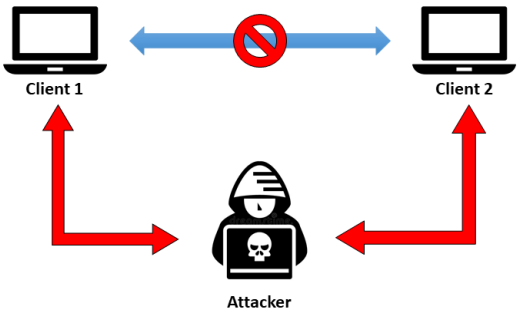
Transport Layer Security (TLS) adalah protokol yang digunakan untuk mengamankan komunikasi antara klien dan server melalui jaringan. TLS dirancang untuk dapat mencegah penyadapan, gangguan dan pemalsuan pesan untuk aplikasi klien-server. Protokol TLS atau biasa disebut SSL (*Secure Sockets Layer*) merupakan protokol yang paling banyak digunakan di internet dengan menyediakan otentikasi, integritas dan kerahasiaan untuk dua pihak. TLS dapat mengamankan informasi rahasia seperti nama pengguna, email dan kata sandi (Turner, 2014). TLS menggunakan *cipher suite* untuk memberikan keamanan komunikasi pada TCP/IP. Sebuah *cipher suite* terdiri dari algoritma kriptografi yang mendukung proses pertukaran kunci, enkripsi dan pengamanan integritas serta keaslian melalui kode otentikasi pesan berupa alamat MAC. Dengan demikian, TLS dapat mengimplementasikan lapisan *transport* terenkripsi antara dua perangkat yang berkomunikasi secara langsung(Prantl dkk., 2021).

2.4. JSON Web Token

JSON Web Token (JWT) merupakan standar format untuk mengamankan informasi pribadi menjadi sebuah klaim yang akan di *encode* ke dalam bentuk JSON dan menjadi *payload* dari JSON Web Signature (JWS). Klaim tersebut dilindungi oleh tanda tangan digital seperti *Message Authentication Code* (MAC) dan juga dapat dienkripsi. JWT merupakan mekanisme autentikasi yang bersifat *stateless* karena *state* dari pengguna yang melakukan login tidak akan disimpan pada *memory* server. Jadi setiap *request* kepada API yang terproteksi akan diperiksa apakah token JWT yang ada di *Authorization header* valid atau tidak. Jika valid maka *request* akan diijinkan dan diproses. (Jones, Bradley & Sakimura, 2013).

2.5. Serangan *Man In The Middle*

Man in the middle merupakan sebuah serangan yang bertujuan untuk membuat protokol autentikasi dimana penyerangan dapat menempatkan dirinya di antara 2 pihak yaitu pemohon autentikasi dan pihak yang bertanggung jawab untuk memverifikasi identitas sehingga penyerang dapat membaca, mencegat dan mengubah data atau pesan yang dikirimkan (Tareq, 2021)(Setiyadi, 2017). Berikut ini merupakan ilustrasi pengiriman data setelah dilakukan serangan *man in the middle* yang dapat dilihat pada gambar dibawah ini.



Gambar 2.1 Ilustrasi Serangan *Man In The Middle*

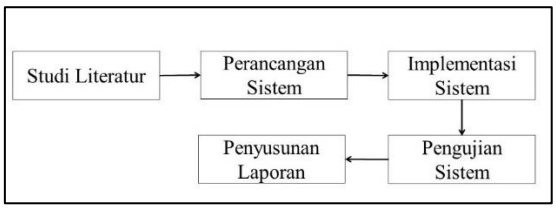
2.6. Mosquitto

Mosquitto merupakan aplikasi yang dapat menyediakan layanan MQTT *broker*. Mosquitto ditulis dalam bahasa C yang membuatnya sangat cepat dan cocok untuk digunakan oleh semua perangkat mulai dari komputer dengan papan *lower power single* hingga ke server. Mosquitto bersifat *open source* yang artinya dapat diakses oleh siapapun secara gratis. Perangkat lunak ini dapat menyediakan autentikasi dan otorisasi dengan plugin keamanan yang dinamis dengan dukungan pada banyak *platform* seperti Cedalo cloud, Docker, lingkungan *cloud* dan lain-lain (Vannebäck, 2018).

3. METODE PENELITIAN

Penelitian ini menggunakan pendekatan kuantitatif dan pengamanan sistem IoT dari serangan *man in the middle* sebagai objek penelitian dengan membuat suatu rancang bangun sistim keamanan untuk perangkat IoT pada

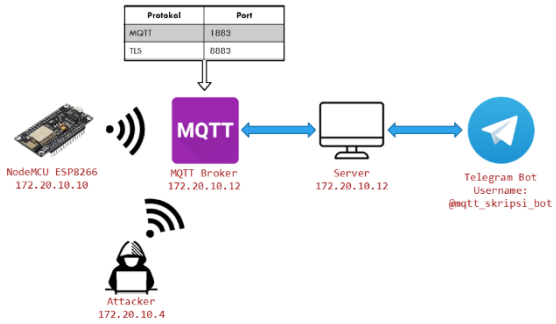
protokol MQTT dengan menerapkan metode autentikasi. Penelitian kuantitatif ini lebih menekankan pada bagaimana sistem yang telah dibangun dapat berjalan sesuai dengan yang diinginkan dengan melakukan pengujian fungsional, pengujian performansi & resistansi sistem terhadap serangan *man in the middle*. Gambar dibawah ini menunjukkan tahapan penelitian yang dilakukan dimulai dari studi literatur, perancangan sistem, implementasi sistem, pengujian sistem hingga penyusunan laporan.



Gambar 3.1 Tahapan Penelitian

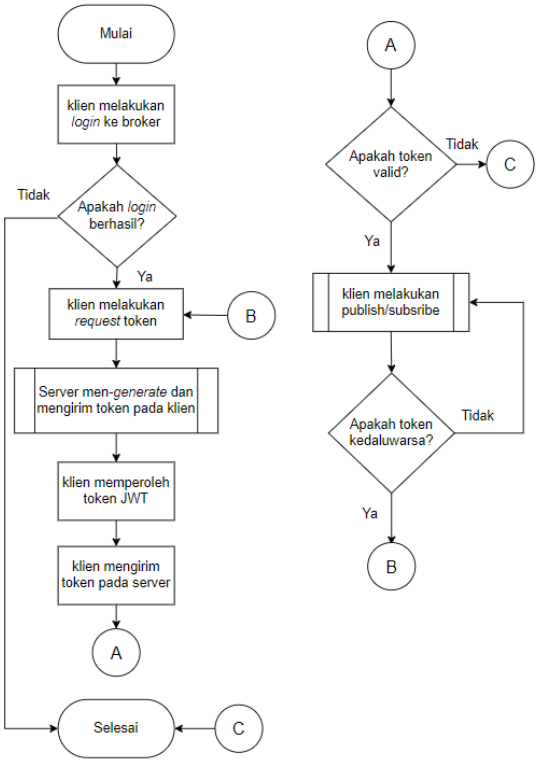
4. HASIL DAN PEMBAHASAN

Penelitian ini berfokus pada rancang bangun sistim keamanan perangkat IoT dengan menerapkan metode autentikasi. Perangkat IoT yang digunakan adalah NodeMCU ESP8266 dengan menggunakan autentikasi JSON Web Token berbasis protokol MQTT. Secara umum topologi sistem dapat dilihat pada gambar 4.1.



Gambar 4.1 Topologi Sistem

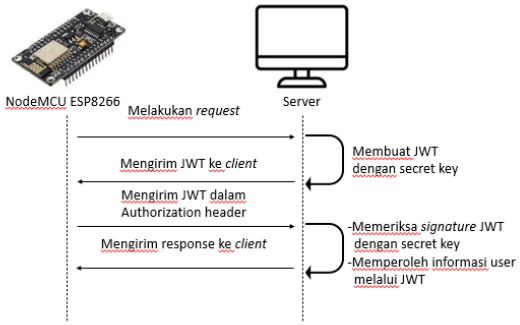
Pada rancangan sistem ini, terdapat empat komponen utama yang terdiri dari NodeMCU ESP8266, MQTT *broker*, server dan aplikasi Telegram sebagai media notifikasi jika terjadi upaya penyerangan berupa *man in the middle* oleh penyerang dengan mendeteksi token JWT yang memiliki tanda tangan *invalid*. NodeMCU ESP8266 dan server saling berkomunikasi menggunakan protokol MQTT. Setelah berhasil terhubung dengan broker NodeMCU ESP8266 sebagai klien akan mengirimkan *request token* kepada server. Kemudian server akan melakukan verifikasi terhadap seluruh token yang diterima. Alur keseluruhan yang menjelaskan urutan mulai dari proses klien terkoneksi dengan *broker* hingga server autentikasi berhasil mendeteksi adanya pengguna tidak terautentikasi dapat dilihat pada gambar 4.2.



Gambar 4.2 Diagram Alir Sistem

4.1 RANCANGAN SISTEM AUTENTIKASI

Sistem autentikasi pada penelitian ini dirancang untuk dilakukan oleh dua pihak yaitu NodeMCU ESP8266 yang akan berperan sebagai klien yang akan menggunakan token JWT dan server yang berperan sebagai pembuat JWT. Alur dan rancangan sisten autentikasi menggunakan JWT dapat dilihat pada gambar 4.3.



Gambar 4.3 Rancangan Sistem Autentikasi JWT

4.2 RANCANGAN SISTEM KOMUNIKASI

Dalam komunikasi sistem selama proses autentikasi

berlangsung, pertukaran data dilakukan dengan protokol MQTT menggunakan skema komunikasi *peer-to-peer*, dimana NodeMCU ESP8266 dan server akan melakukan *publish* dan *subscribe* disaat yang bersamaan. Proses *publish* akan dilakukan didalam fungsi *callback* yang akan merespon setiap pesan yang diterima oleh *subscriber*, sedangkan pada proses *subscribe* terdapat mekanisme tambahan dalam menerima pesan. Mekanisme tambahan tersebut bertujuan untuk memverifikasi pengirim pesan berdasarkan token JWT yang dimiliki. Hal ini diperlukan karena mekanisme *subscribe* yang umum dapat menerima seluruh pesan yang di *publish* pada topik tertentu, sehingga perlu diberikan penanda unik pada setiap pesan yang dikirimkan.

4.3 PENGUJIAN

Terdapat tiga pengujian yang dilakukan pada sistem ini yaitu pengujian fungsional yang bertujuan untuk mengetahui apakah sistem yang telah dibangun telah memenuhi kebutuhan fungsional yaitu sistem dapat melakukan proses autentikasi pada token JWT yang diterima oleh server dan berjalan pada protokol MQTT, pengujian performansi yang bertujuan untuk mengetahui bagaimana kinerja server dalam menghasilkan token JWT berdasarkan berapa lamanya waktu yang dibutuhkan dan pengujian resistansi yang bertujuan untuk mengetahui apakah sistem memiliki resistansi atau ketahanan terhadap serangan *man in the middle*. Berikut ini merupakan data hasil dari ketiga pengujian yang telah dilakukan

1) Pengujian Fungsional

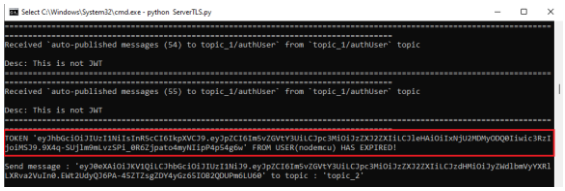
a) Autentikasi dengan token yang valid

Pada gambar 4.4 dapat dilihat bahwa server dapat melakukan autentikasi dan *decode* terhadap token yang diterima. Hasil *decode* tersebut terdiri dari id yang berisi nama pemilik token yaitu ‘nodemcu’, iss yang berisi penerbit atau pembuat token yaitu ‘server’, sts yang berisi status dari token tersebut yaitu satu yang menunjukkan bahwa token yang dikirim oleh server autentikasi telah diterima dan dikonfirmasi oleh penerima token, lalu pada bagian exp menunjukan masa berlaku token tersebut.

Gambar 4.4 Respon Server Terhadap Token Valid

b) Autentikasi dengan token yang kedaluwarsa

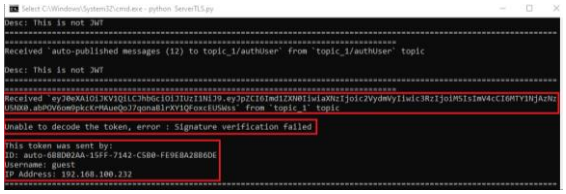
Setiap token yang dibuat oleh server hanya berlaku selama dua puluh detik. Setelah dua puluh detik, token akan kedaluwarsa dan klien harus meminta token baru kepada server. Pada gambar 4.5 menunjukkan bahwa server berhasil melakukan autentikasi terhadap token yang telah kedaluwarsa.



Gambar 4.5 Respon Server Terhadap Token Kedaluwarsa

c) Autentikasi dengan token yang memiliki tanda tangan invalid

Server akan menerima seluruh pesan yang di-publish dan mengenali apakah pesan tersebut merupakan sebuah token. Jika pesan tersebut adalah token, maka server akan melakukan validasi. Apabila token tersebut telah terdeteksi memiliki tanda tangan *invalid* dan memiliki struktur *payload* yang sama dengan token yang dibuat oleh server maka server akan memberikan informasi dan notifikasi melalui aplikasi telegram berupa identias penyerang seperti id, nama pengguna dan alamat ip pengirim token, seperti yang dapat dilihat pada gambar 4.6 dan 4.7.



Gambar 4.6 Respon Server Terhadap Token *Invalid Signature*

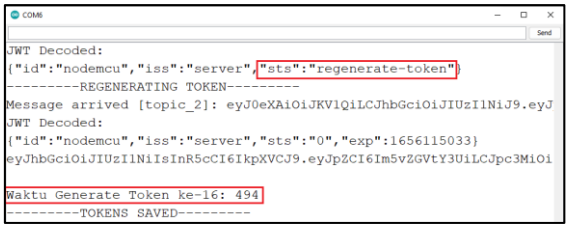
Gambar 4.7 Notifikasi Telegram Berisi Token dengan *Invalid Signature*

Selanjutnya Administrator dapat melakukan blokir terhadap akses dari pengirim token palsu yaitu *attacker* berdasarkan alamat IP yang diperoleh pada notifikasi melalui aplikasi telegram seperti yang dapat dilihat pada gambar 4.8.

Gambar 4.8 Blokir Alamat IP melalui Telegram

2) Pengujian Performansi

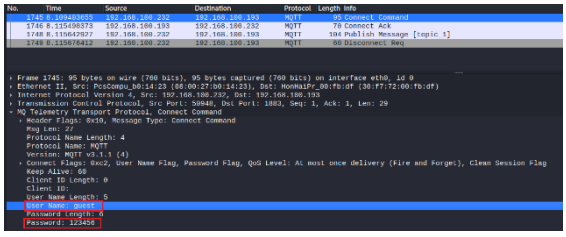
Pengujian performansi dilakukan untuk mengetahui berapa lama waktu yang dibutuhkan oleh klien yaitu NodeMCU ESP8266 untuk dapat memperoleh token JWT dari server. Pengukuran waktu dihitung ketika NodeMCU ESP8266 melakukan request token ke server hingga token tersebut diterima. Pengujian dilakukan sebanyak lima puluh kali. Pada gambar 4.25 menunjukkan bahwa pada salah satu pengujian yang dilakukan yaitu pengujian keenam belas, waktu yang diperlukan NodeMCU ESP8266 untuk memperoleh token adalah 494 millisecond atau 0,494 detik.



Gambar 4.9 Hasil Waktu *Generate* Token pada Salah Satu Pengujian

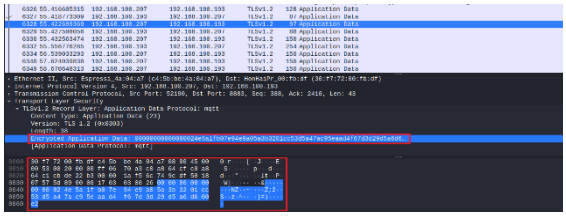
3) Pengujian Resistansi

Pada gambar 4.10 menunjukkan bahwa klien yang melakukan koneksi menggunakan port *default* yaitu 1883 dapat terkena serangan *man in the middle* berupa *sniffing* sehingga informasi penting seperti isi pesan, nama pengguna dan kata sandi dari akun tersebut dapat terlihat dengan jelas dengan menggunakan aplikasi wireshark.



Gambar 4.10 Informasi Akun yang Berhasil Ditangkap oleh Wireshark

Pada gambar 4.11 menunjukkan bahwa klien yang melakukan koneksi dan pertukaran data melalui port 8883 dengan menggunakan protokol keamanan TLS dapat terlindungi dari serangan *man in the middle* berupa *sniffing*. Gambar tersebut menunjukkan bahwa pesan yang berhasil di-*capture* telah terenkripsi sehingga tidak dapat terlihat dengan jelas informasi berupa isi pesan, nama pengguna maupun kata sandi.



Gambar 4.11 Hasil *Capturing* TLS oleh Wireshark

4.4 EVALUASI PENGUJIAN

Setelah pengujian dilakukan, data hasil pengujian dianalisis sesuai dengan masing-masing pengujiannya. Berikut ini merupakan hasil analisis data pengujian.

1) Pengujian Fungsional

Hasil pengujian fungsional yang telah dilakukan pada sistem ini dapat dilihat pada tabel dibawah ini.

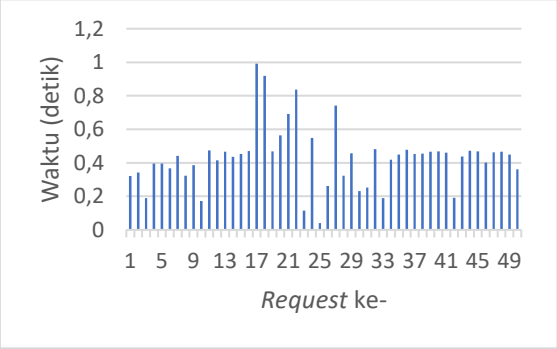
Tabel 4.1 Hasil Pengujian Fungsional

No	Kebutuhan Fungsional	Keterangan (✓)	
		Berhasil	Tidak berhasil
1	Server dapat melakukan autentikasi dengan token yang valid	✓	
2	Server dapat mendeteksi token yang telah kedaluwarsa atau *expired token*	✓	
3	Server dapat mendeteksi token yang memiliki tanda tangan *invalid*	✓	
4	Server dapat mengirimkan notifikasi melalui aplikasi telegram jika terdeteksi adanya token yang memiliki tanda tangan *invalid*	✓	
5	MQTT *broker* dapat melakukan blokir terhadap akses dari alamat IP MQTT klien sesuai dengan *input* yang diberikan oleh administrator melalui aplikasi telegram	✓	

2) Pengujian Performansi

Gambar 4.12 merupakan grafik yang menunjukkan hasil dari pengujian performansi yang telah dilakukan. Pengujian dilakukan sebanyak lima puluh kali dan berjalan menggunakan protokol keamanan TLS. Perhitungan waktu dimulai ketika NodeMCU ESP8266 melakukan *request* token hingga token tersebut diterima. Pada grafik tersebut menunjukkan bahwa waktu paling sedikit yang dibutuhkan yaitu 0,04 detik, waktu yang

paling banyak dibutuhkan yaitu 0,991 detik dan rata-rata waktu yang dibutuhkan oleh NodeMCU ESP8266 untuk memperoleh token JWT berdasarkan hasil pengujian ini yaitu sebesar 0,43028 detik.



Gambar 4.12 Grafik Waktu *Request* Token

3) Pengujian Resistansi

Setelah dilakukan pengujian resistansi serangan *man in the middle* pada sistem ini, maka dapat diketahui bahwa penerapan protokol keamanan TLS berhasil menangkal serangan *man in the middle*. Hal ini dapat dilihat dari pembahasan sebelumnya bahwa setiap pertukaran data yang dilakukan diatas protokol TLS dienkripsi sehingga peretas tidak dapat memperoleh informasi penting dari hasil paket yang telah di-*capture*.

5. PENUTUP

Berdasarkan hasil dari penelitian yang telah dilakukan, maka dapat disimpulkan bahwa:

1. Sistem dapat melakukan autentikasi pada setiap token JWT yang diterima.
2. Sistem dapat mendeteksi token JWT yang memiliki tanda tangan *invalid* dan mengirimkan notifikasi melalui aplikasi Telegram.
3. NodeMCU ESP8266 membutuhkan waktu untuk memperoleh token paling sedikit yaitu 0,04 detik, paling banyak yaitu 0,991 detik dan rata-rata waktu yang dibutuhkan adalah sebesar 0,43028 detik.
4. Sistem yang telah dibangun dapat menangkal serangan *man in the middle* berupa *sniffing* dengan menerapkan protokol keamanan *Transport Layer Security* (TLS) versi 1.2 sehingga mampu mencegah peretas untuk dapat memperoleh informasi penting dari lalu lintas data yang di-*capture*.

6. DAFTAR PUSTAKA

Ahmed, S. & Mahmood, Q. (2019) 'An authentication based scheme for applications using JSON web token', Proceedings - 22nd International Multitopic Conference, INMIC 2019 [Preprint]. doi:10.1109/INMIC48123.2019.9022766.

Bhawiyuga, A., Data, M. & Warda, A. (2018) 'Architectural design of token based authentication of MQTT protocol in constrained IoT device', Proceeding of 2017 11th International Conference on Telecommunication Systems Services and Applications, TSSA 2017, 2018-January, pp. 1–4. doi:10.1109/TSSA.2017.8272933.

Boyd, B. dkk. (2014) Building Real-time mobile solutions with MQTT and IBM MessageSight. IBM Redbooks. Available at: https://www.redbooks.ibm.com/abstracts/sg248228.html?Open (Accessed: 15 March 2022).

Endra, R.Y. dkk. (2019) Smart Room Menggunakan Internet Of Things Untuk Efisiensi Biaya dan Keamanan Ruangan. Bandar Lampung: Aura Publishing. Available at: https://publikasi.ubl.ac.id/index.php/Monograf/catalog/book/34 (Accessed: 30 March 2022).

Fahana, J., Umar, R. & Ridho, F. (2017) 'Pemanfaatan Telegram Sebagai Notifikasi Serangan untuk Keperluan Forensik Jaringan', Query: Journal of Information Systems, 1(2), pp. 6–14. Available at: http://jurnal.uinsu.ac.id/index.php/query/article/view/1036 (Accessed: 26 March 2022).

Jones, M., Bradley, J. & Sakimura, N. (2015) 'JSON Web Token (JWT)', pp. 1–38. Available at: https://www.ietf.org/archive/id/draft-ietf-oauth-json-web-token-13.pdf (Accessed: 15 March 2022).

Kashyap, M., Sharma, V. & Gupta, N. (2018) 'Taking MQTT and NodeMcu to IOT: Communication in Internet of Things', Procedia Computer Science, 132, pp. 1611–1618. doi:10.1016/J.PROCS.2018.05.126.

Kodali, R.K. & Mahesh, K.S. (2016) 'A low cost implementation of MQTT using ESP8266', Proceedings of the 2016 2nd International Conference on Contemporary Computing and Informatics, IC3I 2016, pp. 404–408. doi:10.1109/IC3I.2016.7917998.

Lampkin, V. dkk. (2012) Building Smarter Planet Solutions with MQTT and IBM WebSphere MQ Telemetry, IBM Redbooks. IBM Redbooks. Available at: https://books.google.com/books/about/Building_Smarter_Planet_Solutions_with_M.html?hl=id&id=F_HHAgAAQBAJ (Accessed: 15 March 2022).

Li, S., Xu, L. Da & Zhao, S. (2015) 'The internet of things: a survey', Information Systems Frontiers 2014 17:2, 17(2), pp. 243–259. doi:10.1007/S10796-014-9492-7.

Locke, D. (2010) 'MQ Telemetry Transport (MQTT) V3. 1 Protocol Specification', p. 15.

Lundgren, L. (2016) 'Lightweight Protocol! Serious Equipment! Critical Implications!', in. San Francisco, pp. 1–33.

Mahmoud, R. dkk. (2016) 'Internet of things (IoT) security: Current status, challenges and prospective measures', 2015 10th International Conference for Internet Technology and Secured Transactions, ICITST 2015, pp. 336–341. doi:10.1109/ICITST.2015.7412116.

De Oliveira, J.C., Santos, D.H. & Neto, M.P. (2016) 'Chatting with Arduino platform through Telegram Bot', Proceedings of the International Symposium on Consumer Electronics, ISCE, pp. 131–132. doi:10.1109/ISCE.2016.7797406.

Panjaitan, F. & Syafari, R. (2019) 'PEMANFAATAN NOTIFIKASI TELEGRAM UNTUK MONITORING JARINGAN', Simetris: Jurnal Teknik Mesin, Elektro dan Ilmu Komputer, 10(2), pp. 725–732. doi:10.24176/SIMET.V10I2.3530.

Prantl, T. dkk. (2021) 'Performance Impact Analysis of Securing MQTT Using TLS', ICPE 2021 - Proceedings of the ACM/SPEC International Conference on Performance Engineering, pp. 241–248. doi:10.1145/3427921.3450253.

Setiyadi, A. (2017) 'Implementasi Modul Network MITM Pada Websploit sebagai Monitoring Aktifitas Pengguna dalam Mengakses Internet', Jurnal Teknik Informatika FTIK UNIKOM, pp. 113–120. Available at: https://ojs.unikom.ac.id/index.php/senaski/article/view/934 (Accessed: 30 March 2022).

Shingala, K. (2019) 'JSON Web Token (JWT) based client authentication in Message Queuing Telemetry Transport (MQTT)', pp. 1–14. doi:10.48550/arxiv.1903.02895.

Singh, M. dkk. (2015) 'Secure MQTT for Internet of Things (IoT)', Proceedings - 2015 5th International Conference on Communication Systems and Network Technologies, CSNT 2015, pp. 746–751. doi:10.1109/CSNT.2015.16.

Tareq, M. (2021) 'Man In The Middle Attack in Wireless Network'. Available at: https://www.researchgate.net/publication/356290033_Man_In_The_Middle_Attack_in_Wireless_Network (Accessed: 15 March 2022).

Turner, S. (2014) 'Transport layer security', IEEE Internet Computing, 18(6), pp. 60–63. doi:10.1109/MIC.2014.126.

Vannebäck, E. (2018) 'Using the Mosquitto implementation in an embedded environment', Umeå University, p. 56.

Wahyudi, A. and Suhartati, A. (2016) 'Implementasi Otomatisasi Mesin Grating Menggunakan Mikrokontroler Arduino MEGA 2560', pp. 177–187. Available at: https://journal.untar.ac.id/index.php/tesla/article/view/304/248 (Accessed: 30 March 2022).

Yudha Saputra, G. dkk. (2017) 'Penerapan Protokol MQTT Pada Teknologi Wan (Studi Kasus Sistem Parkir Universitas Brawijaya)', Informatika Mulawarman : Jurnal Ilmiah Ilmu Komputer, 12(2), pp. 69–75. doi:10.30872/JIM.V12I2.653.