BAB 6 PENGUJIAN DAN ANALISIS

6.1 Pengujian Fungsional Sistem

Pengujian Fungsional Sistem merupakan pengujian yang dilakukan untuk menguji fungsionalitas sistem yang sudah diimplementasi. Pengujian fungsional akan dilakukan secara *blackbox* untuk mengetahui apakah sistem sudah berjalan dengan benar serta berjalan dengan baik. Pengujian ini dilakukan dengan melihat hasil interaksi antara klien MQTT, baik itu *subscriber* atau *publisher*, perangkat IoT dalam kasus ini digunakan nodeMCU, Broker MQTT yang dalam kasus ini digunakan mosquitto broker, serta Auth-server yang akan melakukan autentikasi dan otorisasi. Pengujian ini dianggap berhasil jika semua fungsi yang dinyatakan tidak menunjukan kegagalan atau terdapat kesalahan saat dilakukan pengujian.

Setelah melakukan setiap prosedur dalam seluruh skenario pengujian fungsional sistem maka akan dilihat hasil yang didapatkan apakah sistem dapat menanganinya dengan baik dan benar. Nilai keberhasilan pengujian fungsional dinilai dari hasil pengujian skenario apakah terjadi *error* atau kegagalan pada salah satu prosedur di skenario tersebut. Hasil pengujian fungsional sistem telah disimpulkan dan dapat dilihat pada tabel 6.1 berikut.

Kode	Fungsi	Hasil
PFS_001	Klien MQTT dapat membangun koneksi dengan broker MQTT menggunakan token JWT	Berhasil
PFS_002	<i>Publisher</i> dapat melakukan Publish pada suatu topik ke broker MQTT	Berhasil
PFS_003	Subscriber dapat melakukan Subscribe pada suatu topik ke broker MQTT	Berhasil
PFS_004	Auth-server dapat melakukan mekanisme autentikasi dengan memeriksa identitas klien MQTT pada database.	Berhasil
PFS_005	Auth-server dapat melakukan mekanisme otorisasi dengan memeriksa ACL terkait hak akses klien MQTT pada database.	Berhasil
PFS_006	Auth-server dapat melakukan <i>generate</i> token JWT berdasakan username dan password pengguna yang ada di database	Berhasil
PFS_007	Admin dapat melakukan <i>Create User</i> untuk membuat sebuah pengguna baru lewat auth-server	Berhasil
PFS_008	Admin dapat melakukan <i>Update User</i> untuk mengubah pengguna lewat auth-server	Berhasil
PFS_009	Admin dapat melakukan <i>Delete User</i> untuk menghapus pengguna lewat auth-server	Berhasil
PFS_010	Admin dapat melakukan <i>Create Role</i> untuk membuat peran baru lewat auth-server	Berhasil

Tabel 6.1 Hasil Pengujian Fungsional Sistem

PFS_011	Admin dapat melakukan <i>Delete Role</i> untuk menghapus peran lewat auth-server	Berhasil
PFS_012	Admin dapat melakukan <i>Create Permission</i> untuk membuat perizinan baru lewat auth-server	Berhasil
PFS_013	Admin dapat melakukan <i>Delete Permission</i> untuk menghapus perizinan lewat auth-server	Berhasil
PFS_014	NodeMCU + LED berhasil terhubung dengan broker serta melakukan <i>subscribe</i> dan melakukan <i>publish</i>	Berhasil
PFS_015	NodeMCU + DHT11 berhasil terhubung dengan broker serta membaca data dan melakukan <i>publish</i>	Berhasil

6.2 Pengujian Keamanan Sistem

Pengujian Keamanan Sistem merupakan pengujian yang dilakukan untuk mengetahui apakah sistem yang telah berhasil dibuat mampu melakukan mekanisme autentikasi menggunakan token JWT dan mekanisme otorisasi menggunakan access control list (ACL) untuk mengatur hak akses melalui auth-server di sistem berbasis protokol MQTT berhasil dilakukan. Pengujian ini juga dilakukan untuk mengetahui perbedaan sebelum dan sesudah penerapan auth-server sebagai pihak yang melakukan mekanisme autentikasi dan otorisasi pada mosquitto broker. Pengujian keamanan akan dilakukan dengan cara melakukan sniffing pada setiap paket data yang dikirimkan dari dan menuju broker menggunakan aplikasi Wireshark serta akan dilihat bagaimana kondisi dari broker pada transaksi setiap paket data tersebut.

6.2.1 Pengujian Mekanisme Autentikasi Sistem

Pengujian ini dilakukan untuk mengetahui hasil penerapan mekanisme autentikasi pada sistem berbasis protokol MQTT dengan menggunakan token JWT sebagai parameter autentikasi. Pengujian ini akan membandingkan bagaimana perbedaan keamanan pada ketiga konfigurasi broker MQTT yang sudah dijabarkan, ketiga konfigurasi broker akan memiliki beberapa skenario yang akan dilakukan. Pengujian akan dilakukan dengan cara klien MQTT akan mengirimkan pesan *Connect* untuk membangun koneksi ke broker MQTT, kemudian pada tiap skenario akan digunakan parameter yang berbeda, apakah dengan token JWT, *username* dan *password* atau tanpa keduanya.

Pada pengujian ini, pertama-tama Wireshark akan dijalankan dan melakukan *capture* kemudian klien MQTT akan mengirimkan pesan *connect* ke broker MQTT untuk mencoba membangun koneksi. Pesan *connect* akan dikirimkan menggunakan aplikasi dashboard MQTT-spy dengan parameter *user credentials* yang diseusaikan dengan skenario, setelah dilakukan pengiriman pesan *connect* dari klien selanjutnya akan dilihat hasil *capture* dari Wireshark dan status dari mosquitto-broker dan juga auth-server jika digunakan.

6.2.1.1 Klien Membangun Koneksi Dengan Broker Tanpa Menggunakan Username Dan Password. (Kode: PMO_101)

Pada skenario pengujian ini digunakan konfigurasi broker MQTT yang tidak menerapkan Auth-Server dan tidak menerapkan TLS. Prosedur pengujian pada skenario ini akan dimulai dengan melakukan *capture interfaces* di Wireshark, lalu dengan aplikasi MQTT-Spy klien akan mengirim pesan *CONNECT* ke broker tanpa menyertakan *username* dan *password*, setelah itu akan dilihat status broker, selanjutnya akan dilakukan *filter* pada hasil *capture interface* menggunakan *field* MQTT yang akan menyeleksi setiap paket yang dikirimkan menggunakan protokol MQTT untuk melihat hasil *sniffing*.

```
ardhiaan@ardhiaan-VirtualBox:~$ mosquitto -v -c mqtt-mosquitto-conf/mosquitto-no
-auth.conf
1509616149: mosquitto version 1.3.5 (build date 2017-05-17 13:25:43+0700) starti
ng
1509616149: Config loaded from mqtt-mosquitto-conf/mosquitto-no-auth.conf.
1509616149: Opening ipv4 listen socket on port 1883.
1509616149: Opening ipv6 listen socket on port 1883.
1509616223: New connection from 192.168.100.29 as Client_Pengujian (c1, k6
0).
1509616223: Sending CONNACK to Client_Pengujian (0)
```

Gambar 6.1 Screenshot Broker PMO_101

Pada gambar 6.1, dapat dilihat bahwa broker menerima permintaan koneksi dari klien dan mengirimkan pesan CONNACK bernilai 0, yang menandakan peermintaan koneksi diterima, dengan menggunakan konfigurasi ini klien yang bersifat anonim atau klien tanpa *username* dan *password* dapat membangun koneksi dengan broker dan mengakses sistem, sehingga sistem sangat *vulnerable* terhadap penyalahgunaan informasi yang tersedia di dalamnya karena siapapun dapat mengakses sistem tanpa memerlukan *username* dan *password*.

8	🕒 🗉 *any				
<u> </u>	e <u>E</u> dit <u>V</u> iew <u>G</u> o <u>C</u>	apture <u>A</u> nalyze	Statistics Telephony Wir	eless <u>T</u> ools	<u>H</u> elp
		i 🖹 🖹 🎑	۹ 🗭 🛸 🖀 有	₹	€ € € 1
	mqtt				
No.	Time	Source	Destination	Protocol	Length Info
	6 0.034667767 8 0.035032226 14 60.068247469 15 60.068371626 46 120.057361673 47 120.057690146 28955 180.052823087 28956 180.052934121	$\begin{array}{r} 192.168.100.29\\ 192.168.100.40\\ 192.168.100.29\\ 192.168.100.40\\ 192.168.100.29\\ 192.168.100.40\\ 192.168.100.40\\ 192.168.100.40\\ 192.168.100.40\\ \end{array}$	$\begin{array}{r} 192.168.100.40\\ 192.168.100.29\\ 192.168.100.40\\ 192.168.100.29\\ 192.168.100.29\\ 192.168.100.29\\ 192.168.100.29\\ 192.168.100.40\\ 192.168.100.40\\ 192.168.100.29\end{array}$	MQTT MQTT MQTT MQTT MQTT MQTT MQTT	86 Connect Command 60 Connect Ack 62 Ping Request 58 Ping Response 62 Ping Request 58 Ping Response 62 Ping Request 58 Ping Response
-	4Q Telemetry Transport Connect Command > 0001 0000 = He Msg Len: 28 Protocol Name: Version: 4 > 0000 0010 = Co Keep Alive: 60 Client ID: Client	ort Protocol eader Flags: 0x10 MQTT onnect Flags: 0x0 ont Penguijan	(Connect Command) 2		

Gambar 6.2 Hasil Capture Wireshark PMO_101

Seperti yang terlihat pada gambar 6.2, bahwa hasil *sniffing* protokol MQTT mendapatkan pesan *CONNECT* yang dikirimkan oleh klien ke broker. Jika dilihat pada isi data dari paket *CONNECT* tersebut dapat dilihat bahwa data yang dikirimkan dapat dibaca dalam bentuk *plain text*, karena dalam kasus ini tidak digunakan *username* dan *password* maka data yang ada dalam paket ini hanya "client_ID" yang memiliki nilai "Client_Pengujian".

6.2.1.2 Klien Membangun Koneksi Dengan Broker Dengan Menggunakan *Username* Dan *Password*. (Kode: PMO_102)

Pada skenario pengujian ini digunakan konfigurasi broker MQTT yang tidak menerapkan Auth-Server dan tidak menerapkan TLS. Prosedur pengujian pada skenario ini akan dimulai dengan melakukan *capture interfaces* di Wireshark, lalu dengan aplikasi MQTT-Spy klien akan mengirim pesan *CONNECT* ke broker dengan menyertakan *username* dan *password* yang digunakan secara sembarangan, setelah itu akan dilihat status broker, selanjutnya akan dilakukan *filter* pada hasil *capture interface* menggunakan *field* MQTT yang akan menyeleksi setiap paket yang dikirimkan menggunakan protokol MQTT untuk melihat hasil *sniffing*.

ardhiaan@ardhiaan-VirtualBox:~\$ mosquitto -v -c mqtt-mosquitto-conf/mosquitto-no
-duth.com
ng
1509616845: Config loaded from mqtt-mosquitto-conf/mosquitto-no-auth.conf.
1509616845: Opening ipv4 listen socket on port 1883.
1509616845: Opening ipv6 listen socket on port 1883.
1509616856: New connection from 192.168.100.29 on port 1883.
1509616856: New client connected from 192.168.100.29 as Client Pengujian (c1, k6
0, uTest username).
1509616856: Sending CONNACK to Client_Pengujian (0)

Gambar 6.3 Screenshot Broker PMO_102

Pada gambar 6.3, dapat dilihat bahwa broker menerima permintaan koneksi dari klien dan mengirimkan pesan CONNACK bernilai 0, yang menandakan peermintaan koneksi diterima, dengan menggunakan konfigurasi ini klien dapat melakukan koneksi dengan broker dengan menggunakan *username* dan *password* yang digunakan secara asal. Berdasarkan hasil ini diketahui, penggunaan *username* dan *password* pada konfigurasi ini tidak berpengaruh sama sekali dan memiliki *vulnerable* terhadap penyalahgunaan informasi yang tersedia di dalamnya, sama seperti pada skenario sebelumnya.

Pada gambar 6.4, dapat dilihat bahwa hasil *sniffing* protokol MQTT mendapatkan pesan *CONNECT* yang dikirimkan oleh klien ke broker. Jika dilihat pada isi data dari paket *CONNECT* tersebut dapat dilihat bahwa data yang dikirimkan dapat dibaca dalam bentuk *plain text*, dalam kasus ini digunakan *username* dan *password* yang digunakan secara asal sehingga dapat dilihat bahwa data yang ada dalam paket ini adalah *Client* ID yang memiliki nilai Client_Pengujian, *User Name* yang memiliki nilai "Test_Username" dan *Password* yang memiliki nilai "test1234".

8	🔲 *any						
<u>F</u> ile	<u>E</u> dit <u>V</u> iew	<u>G</u> o <u>C</u> aptu	re <u>A</u> nalyze	Statistics Telephon	<u>y W</u> ireless <u>T</u>	ools <u>H</u> elp	
	I 🔊		XC	ې 🗢 م	T 🖢	•	Q Q 🏢
🔳 m	qtt						
No.	Time	Sour	ce	Destination	Proto	col Length Info	
	6 0.00197 10 60.0042 11 60.0044	7088 192 264710 192 38519 192	168.100.40 168.100.29 168.100.40	192.168.100. 192.168.100. 192.168.100.	29 MQTT 40 MQTT 29 MQTT	60 Con 62 Pin 58 Pin	nect Ack g Request g Response
▼ MQ	Telemetry Connect Co 0001 000 Msg Len Protoco Version 1100 000 Keep Al Client S User Nan Password	Transport F mmand 00 = Header : 53 1 Name: MQT : 4 10 = Connec ive: 60 ID: Client_ me: Test_us d: test1234	Protocol Flags: 0x1 T t Flags: 0x Pengujian ername	0 (Connect Command) c2)		

Gambar 6.4 Hasil Capture Wireshark PMO_102

6.2.1.3 Klien Membangun Koneksi Dengan Broker Tanpa Menggunakan Token JWT. (Kode: PMO_201)

Pada skenario pengujian ini digunakan konfigurasi broker MQTT yang akan menerapkan Auth-Server menggunakan *backend* JWT namun tidak menerapkan TLS. Prosedur pengujian pada skenario ini akan dimulai dengan melakukan *capture interfaces* di Wireshark, lalu dengan aplikasi MQTT-Spy klien akan mengirim pesan *CONNECT* ke broker tanpa menyertakan token JWT pada *field username* dan *field password* dapat dikosongkan atau dapat diisi dengan nilai sembarangan, setelah itu akan dilihat status broker, selanjutnya akan dilakukan *filter* pada hasil *capture interface* menggunakan *field* MQTT yang akan menyeleksi setiap paket yang dikirimkan menggunakan protokol MQTT untuk melihat hasil *sniffing*.

1509617711:	New	connection from 192.168.100.29 on port 1883.
1509617711:		<pre>mosquitto_auth_unpwd_check(client1)</pre>
1509617711:		** checking backend jwt
1509617711:		url=http://localhost:8100/auth
1509617711:		data=topic=&acc=-1&clientid=
1509617711:		getuser(client1) AUTHENTICATED=0 by none
1509617711:	Send	ling CONNACK to 192.168.100.29 (4)

Gambar 6.5 Screenshot Broker PMO_201

Pada gambar 6.5, dapat dilihat bahwa broker menolak permintaan koneksi dari klien dan mengirimkan pesan CONNACK bernilai 4, yang menandakan permintaan koneksi ditolak karena tidak terautentikasi, dengan menggunakan konfigurasi ini klien tidak dapat melakukan koneksi dengan broker karena menggunakan format *username* dan *password* yang salah atau tidak menggunakan format sama sekali. Karena mekanisme autentikasi yang dirancang menggunakan token JWT untuk pemeriksaan *username* dan *password*, sehingga akan pemeriksaan *username* dan *password* akan ditolak oleh auth-server yang

mengakibatkan broker menolak permintaan koneksi klien. Berdasarkan hasil ini diketahui, bahwa sistem berhasil menolak permintaan koneksi klien yang anonim atau yang tidak menggunakan token JWT sebagai formatnya.

Pada gambar 6.6, dapat dilihat bahwa hasil *sniffing* protokol MQTT mendapatkan pesan *CONNECT* yang dikirimkan oleh klien ke broker. Jika dilihat pada isi data dari paket *CONNECT* tersebut dapat dilihat bahwa data yang dikirimkan dapat dibaca dalam bentuk *plain text*, dalam kasus ini digunakan *username* dan *password* sehingga dapat dilihat bahwa data yang ada dalam paket ini adalah *Client* ID yang memiliki nilai "Client_Pengujian", *User Name* yang memiliki nilai "client1" dan *Password* yang memiliki nilai "contoh12".



Gambar 6.6 Hasil Capture Wireshark PMO_201

6.2.1.4 Klien Membangun Koneksi Dengan Broker Dengan Menggunakan Token JWT yang *Invalid*. (Kode: PMO_202)

Pada skenario pengujian ini digunakan konfigurasi broker MQTT yang akan menerapkan Auth-Server menggunakan *backend* JWT namun tidak menerapkan TLS. Prosedur pengujian pada skenario ini akan dimulai dengan melakukan *capture interfaces* di Wireshark, lalu dengan aplikasi MQTT-Spy klien akan mengirim pesan *CONNECT* ke broker dengan menyertakan token JWT yang *invalid*, yaitu token JWT milik pengguna yang tidak terdaftar dalam database pada *field username*, dan *field password* dapat dikosongkan atau dapat diisi dengan nilai sembarangan, setelah itu akan dilihat status broker, selanjutnya akan dilakukan *filter* pada hasil *capture interface* menggunakan *field* MQTT yang akan menyeleksi setiap paket yang dikirimkan menggunakan protokol MQTT untuk melihat hasil *sniffing*.

Pada gambar 6.7, dapat dilihat bahwa broker berhasil menolak permintaan koneksi dari klien dan mengirimkan pesan CONNACK bernilai 4, yang menandakan permintaan koneksi ditolak karena tidak terautentikasi, dengan menggunakan

konfigurasi ini klien dapat melakukan koneksi dengan broker karena menggunakan format *username* dengan token JWT dan *password* yang akan diabaikan nilainya, namun karena isi dari *payload* token JWT merupakan data yang tidak terdaftar dalam database maka permintaan koneksi akan ditolak. Karena mekanisme autentikasi yang dirancang menggunakan token JWT untuk pemeriksaan *username* dan *password*, sehingga pemeriksaan *username* dan *password* akan didapatkan dari *payload* pada token JWT oleh auth-server, lalu akan diperiksa di database dan auth-server gagal menemukan data klien pada database yang mengakibatkan broker untuk menolak permintaan koneksi klien. Berdasarkan hasil ini diketahui, bahwa sistem berhasil memeriksa token JWT dengan data klien pada database dan menolak permintaan koneksi klien yang tidak terdaftar dalam sistem.



Gambar 6.7 Screenshot Broker PMO_202

Pada gambar 6.8, dapat dilihat bahwa hasil *sniffing* protokol MQTT mendapatkan pesan *CONNECT* yang dikirimkan oleh klien ke broker. Jika dilihat pada isi data dari paket *CONNECT* tersebut dapat dilihat bahwa data yang dikirimkan dapat dibaca dalam bentuk *plain text*, dalam kasus ini karena digunakan token JWT sebagai *field username* sehingga informasi klien dalam bentuk *encode* JWT, sedangkan *password* dapat dibaca secara *plain text* namun karena *password* ini akan diabaikan dan *field* password yang akan digunakan adalah yang terdapat dalam *payload* token JWT. Meskipun informasi *username* dan *password* klien terlindungi oleh *encode* token JWT namun token JWT ini dapat diambil nilainya dan dapat digunakan untuk mengakses sistem sebagai klien ini, sehingga masih dibutuhkan faktor keamanan tambahan.

8	🕒 *any								
File	<u>E</u> dit <u>V</u> iew <u>G</u> o	<u>Capture</u> <u>A</u> nalyze <u>S</u> ta	itistics Telephon <u>y W</u>	ireless <u>T</u> ools	<u>H</u> elp				
		🚞 🖹 🖹 🍯	🔪 🗢 🔿 🖉 有	· 👱 📃 🛛	€ €	♀. ፹			
m	qtt						Expression	. + +	
No.	Time	Source	Destination	Protocol I	engtr Info				-
	6 0.380576825	192.168.100.29	192.168.100.40	MQTT	224 Connect	Command			
	27 1.522779091	192.168.100.40	192.168.100.29	MQTT	60 Connect	Ack			
	35 1.749918119	192.168.100.29	192.168.100.40	MQTT	226 Connect	Command			
	53 1 756621685	192 168 100 40	192 168 100 29	MOTT	60 Connect	Ack		_	
•	Connect Comman	d							
	▶ 0001 0000 =	Header Flags: 0x10 (Connect Command)						
	Msg Len: 165								
	Protocol Nam	ie: MQII							
	Version: 4	Connect Flags: 0xe2							
	Koop Alivo:	60							
	Client ID: (lient Denguiian							
	llear Name: a	vlbbGciOilTUzT1NiTeT	PScCT6TknXVC10 ev11	c2V/vbmE+7ST6T	mEzVWwilClwdv?	T6TmEhVS10 kihGh7W	r - BeWeGd0D0VdBvcCXRNK	/601.1	4
	Dassword: na	sswordAsal	Nocololkby/0000.6901	02 V Y DIII - C 20101	IIF21WW1L00Wdy1	IOIMPITO55. KEHODIW	1-D346000001004000000	DOLL	
	rassworu, po	135WOT GASKE							-
4								Þ	-

Gambar 6.8 Hasil Capture Wireshark PMO_202

6.2.1.5 Klien Membangun Koneksi Dengan Broker Dengan Menggunakan Token JWT yang *Valid*. (Kode: PMO_203)

Pada skenario pengujian ini digunakan konfigurasi broker MQTT yang akan menerapkan Auth-Server menggunakan *backend* JWT namun tidak menerapkan TLS. Prosedur pengujian pada skenario ini akan dimulai dengan melakukan *capture interfaces* di Wireshark, lalu dengan aplikasi MQTT-Spy klien akan mengirim pesan *CONNECT* ke broker dengan menyertakan token JWT yang *valid*, yaitu token JWT milik pengguna yang terdaftar dalam database pada *field username*, dan *field password* dapat dikosongkan atau dapat diisi dengan nilai sembarangan, setelah itu akan dilihat status broker, selanjutnya akan dilakukan *filter* pada hasil *capture interface* menggunakan *field* MQTT yang akan menyeleksi setiap paket yang dikirimkan menggunakan protokol MQTT untuk melihat hasil *sniffing*.

Pada gambar 6.9, dapat dilihat bahwa broker berhasil menerima permintaan koneksi dari klien dan mengirimkan pesan CONNACK bernilai 0, yang menandakan permintaan koneksi diterima, dengan menggunakan konfigurasi ini klien dapat melakukan koneksi dengan broker karena menggunakan format *username* dengan token JWT dan *password* yang akan diabaikan nilainya. Karena mekanisme autentikasi yang dirancang menggunakan token JWT untuk pemeriksaan *username* dan *password*, sehingga pemeriksaan *username* dan *password* akan didapatkan dari *payload* pada token JWT oleh auth-server, lalu akan diperiksa di database dan auth-server berhasil menemukan data klien pada database yang mengakibatkan broker menerima permintaan koneksi klien. Berdasarkan hasil ini diketahui, bahwa sistem berhasil memeriksa token JWT dengan data klien pada database dan menerima permintaan koneksi klien yang terdaftar dalam sistem.

1509618109: New connection from 192.168.100.29 on port 1883.
1509618109: mosquitto_auth_unpwd_check(eyJhbGci0iJIUzI1NiIsInR5cCI6IkpXVCJ9.
eyJ1c2VybmFtZSI6ImNsaWVudDEiLCJwdyI6ImNvbnRvaDEyIn0.043ve3RXXPuYm0QVnc4CF-6uxT-8
_TU2ij70cqTKNME)
1509618109: ** checking backend jwt
1509618109: url=http://localhost:8100/auth
1509618109: data=topic=&acc=-1&clientid=
1509618109: getuser(eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmFtZSI6ImN
saWVudDEiLCJwdyI6ImNvbnRvaDEyIn0.043ve3RXXPuYm0QVnc4CF-6uxT-8_TU2ij7OcqTKNME) AU
THENTICATED=1 by jwt
1509618109: New client connected from 192.168.100.29 as Client_Pengujian (c1, k6
0, ueyJhbGci0iJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmFtZSI6ImNsaWVudDEiLCJwdyI6ImN
vbnRvaDEyIn0.043ve3RXXPuYm0QVnc4CF-6uxT-8_TU2ij7OcqTKNME).
1509618109: Sending CONNACK to Client Penguilan (0)

Gambar 6.9 Screenshot Broker PMO_203

Pada gambar 6.10, dapat dilihat bahwa hasil *sniffing* protokol MQTT mendapatkan pesan *CONNECT* yang dikirimkan oleh klien ke broker. Jika dilihat pada isi data dari paket *CONNECT* tersebut dapat dilihat bahwa data yang dikirimkan dapat dibaca dalam bentuk *plain text*, dalam kasus ini karena digunakan token JWT sebagai *field username* sehingga informasi klien dalam bentuk *encode* JWT, sedangkan *password* dapat dibaca secara *plain text* namun karena *password* ini akan diabaikan dan *field* password yang akan digunakan adalah yang terdapat dalam *payload* token JWT. Meskipun informasi *username* dan *password* klien terlindungi oleh *encode* token JWT namun token JWT ini dapat

diambil nilainya dan dapat digunakan untuk mengakses sistem sebagai klien ini, sehingga masih dibutuhkan faktor keamanan tambahan. Hasil yang didapatkan dari *sniffing* paket MQTT pada skenario ini sama dengan yang ada pada skenario PMO_202.



Gambar 6.10 Hasil Capture Wireshark PMO_203

6.2.1.6 Klien Membangun Koneksi Dengan Broker Tanpa Menggunakan Token JWT. (Kode: PMO_301)

Pada skenario pengujian ini digunakan konfigurasi broker MQTT yang akan menerapkan Auth-Server menggunakan *backend* JWT dan menerapkan keamanan TLS. Prosedur pengujian pada skenario ini akan dimulai dengan melakukan *capture interfaces* di Wireshark, lalu dengan aplikasi MQTT-Spy klien akan mengirim pesan *CONNECT* ke broker tanpa menyertakan token JWT pada *field username* dan *field password* dapat dikosongkan atau dapat diisi dengan nilai sembarangan, kemudian akan digunakan *Certificate of Authority* (CA) yang dibuat menggunakan OpenSSL. Setelah itu akan dilihat status broker, selanjutnya akan dilihat hasil *capture interface* terkait setiap paket yang dikirimkan berdasarkan alamat ip klien yang dikirim menuju alamat ip milik broker MQTT untuk melihat hasil *sniffing* di TLS.

1509620906:	New connection from 192.168.100.29 on port 8883.
1509620906:	<pre> mosquitto_auth_unpwd_check(client1)</pre>
1509620906:	** checking backend jwt
1509620906:	url=http://localhost:8100/auth
1509620907:	<pre> data=topic=&acc=-1&clientid=</pre>
1509620907:	<pre> getuser(client1) AUTHENTICATED=0 by none</pre>
1509620907:	Sending CONNACK to 192.168.100.29 (4)

Gambar 6.11 Screenshot Broker PMO_301

Pada gambar 6.11, dapat dilihat bahwa broker menolak permintaan koneksi dari klien dan mengirimkan pesan CONNACK bernilai 4, yang menandakan permintaan koneksi ditolak karena tidak terautentikasi, dengan menggunakan konfigurasi ini klien tidak dapat melakukan koneksi dengan broker karena menggunakan format *username* dan *password* yang salah atau tidak menggunakan format sama sekali. Karena mekanisme autentikasi yang dirancang menggunakan token JWT untuk pemeriksaan *username* dan *password*, sehingga

akan pemeriksaan username dan password akan ditolak oleh auth-server yang mengakibatkan broker menolak permintaan koneksi klien. Berdasarkan hasil ini diketahui, bahwa sistem berhasil menolak permintaan koneksi klien yang anonim atau yang tidak menggunakan token JWT sebagai formatnya. Hasil yang didapatkan oleh broker pada skenario ini dama dengan yang dilakukan pada skenario PMO_201, karena mekanisme autentikasi yang dibuat pada auth-server tidak diubah.

i	8 🔿 🤇	🗈 *any																
Γ	<u>F</u> ile <u>E</u>	dit <u>V</u> iew	<u>G</u> o	<u>C</u> apture	<u>A</u> nalyze	Statistic	s Te	lephony	<u>W</u> irele	ss <u>T</u> ools	6 <u>H</u> elp							
			۲	01010	8	9)) 🖄	<u></u>			Ð	⊜	8				
(App	ly a display	y filter .	<ctrl- <="" th=""><th>></th><th></th><th></th><th></th><th></th><th></th><th></th><th></th><th></th><th></th><th></th><th></th><th>Express</th><th>ion</th></ctrl->	>												Express	ion
Ī	No.	Time		Source		D	estina	tion		Protocol	Length	Info						
		7 12.1765 8 12.1766 9 12.1769	581365 523778 963189	192.16 192.16 192.16	8.100.29 8.100.40 8.100.29	1 1 1	92.16 92.16 92.16	8.100.4 8.100.2 8.100.4	0 9 0	TCP TCP TCP	68 68 62	49239 8883 → 49239	→ 8883 49239 → 8883	[SYN] [SYN, [ACK]	Seq=0 W ACK] Se Seq=1 A	in=64240 q=0 Ack=1 ck=1 Win=	Len=0 MS Win=292 525568 L	S=1 00 en=0
	4	12.178 1 12.178 2 12.1792	776425 803480 211513	192.16 192.16 192.16	8.100.29 8.100.40 8.100.40	1	92.16 92.16 92.16	8.100.4 8.100.2 8.100.2	9 9	TLSV1.2 TCP TLSV1.2	250 56 1128	Client 8883 → Server	Hello 49239 Hello,	[ACK] Cert	Seq=1 A ificate,	ck=195 Wi Server H	n=30336 ello Don	Len e
		24 12.2220 25 12.2229 26 12.2229	699315 939889 963085	192.10 192.16 192.16 192.16	8.100.29 8.100.29 8.100.29 8.100.40	1	92.10 92.16 92.16 92.16	8.100.2 8.100.4 8.100.2	9 0 9	TLSv1.2 TCP TLSv1.2 TCP	56 107 56	8883 → Change 8883 →	49239 Cipher 49239 Cipher	[ACK] Spec [ACK]	Seq=107 Hello Seq=107	3 Ack=462 Request, 3 Ack=513	Win=313 Hello Re Win=313	60 que 60
Ì		8 12.2262	209942	192.10	8.100.29	1	92.10	8.100.2	0	TLSv1.2	138	Applic	ation [ata	, спотур	ceu nanus	nake nes	saye
	 Frame 28: 138 bytes on wire (1104 bits), 138 bytes captured (1104 bits) on interface 0 Linux cooked capture Internet Protocol Version 4, Src: 192.168.100.29, Dst: 192.168.100.40 Transmission Control Protocol, Src Port: 49239, Dst Port: 8883, Seq: 513, Ack: 1124, Len: 82 Secure Sockets Layer 																	
	▼ T	LSv1.2 Re Content Version Length:	Cord L Type: TLS 77	ayer: A Applica 1.2 (0x0	pplicatio ation Data 0303)	n Data a (23)	roto	col: mq	tt			0-6450						

Gambar 6.12 Hasil Capture Wireshark PMO_301

Seperti yang terlihat pada gambar 6.12, bahwa hasil *sniffing interface* didapatkan tidak ditemukan protokol MQTT, hal ini karena pertukaran paket data dilakukan melalui TLS. Hasil *sniffing* tidak mendapatkan pesan *CONNECT* yang dikirimkan oleh klien ke broker, namun jika dilihat berdasarkan paket yang dikirim dari ip klien menuju ip broker dapat dilihat pertukaran paket *Application Data* setelah melakukan handshake dan pertukaran sertifikat CA dan *key*. Jika dilihat pada isi data dari paket *Application Data* tersebut dapat dilihat bahwa data yang dikirimkan dapat tidak dapat dibaca karena dalam bentuk enkripsi data.

6.2.1.7 Klien Membangun Koneksi Dengan Broker Dengan Menggunakan Token JWT yang *Invalid*. (Kode: PMO_302)

Pada skenario pengujian ini digunakan konfigurasi broker MQTT yang akan menerapkan Auth-Server menggunakan *backend* JWT dan menerapkan keamanan TLS. Prosedur pengujian pada skenario ini akan dimulai dengan melakukan *capture interfaces* di Wireshark, lalu dengan aplikasi MQTT-Spy klien akan mengirim pesan *CONNECT* ke broker dengan menyertakan token JWT yang *invalid*, yaitu token JWT milik pengguna yang tidak terdaftar dalam database pada *field username*, dan *field password* dapat dikosongkan atau dapat diisi dengan nilai sembarangan, kemudian akan digunakan *Certificate of Authority* (CA) yang dibuat menggunakan OpenSSL. Setelah itu akan dilihat status broker, selanjutnya akan dilihat hasil *capture interface* terkait setiap paket yang dikirimkan berdasarkan alamat ip klien yang dikirim menuju alamat ip milik broker MQTT untuk melihat hasil *sniffing* di TLS.

1509620245: New connection from 192.168.100.29 on port 8883.
1509620245: mosquitto auth unpwd check(eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.,
eyJ1c2VybmFtZSI6ImFzYWwiLCJwdyT6ImFhYSJ9.kLhGb7Wr-BsWeGd0DQYdBvcCXRNKVb9L1HY40H6 L76I)
1509620245: ** checking backend jwt
1509620245: url=http://localhost:8100/auth
1509620245: data=topic=&acc=-1&clientid=
1509620245: getuser(eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmFtZSI6ImF,
zYWwiLCJwdyI6ImFħYSJ9.kLhĠb7Wr-BsWeGd0DQYdBvcCXRNKVb9L1HY4OH6L76I) AUTHENTICATED =0 by none
1509620245: Sending CONNACK to 192.168.100.29 (4)

Gambar 6.13 Screenshot Broker PMO_302

Pada gambar 6.13, dapat dilihat bahwa broker berhasil menolak permintaan koneksi dari klien dan mengirimkan pesan CONNACK bernilai 4, yang menandakan permintaan koneksi ditolak karena tidak terautentikasi, dengan menggunakan konfigurasi ini klien dapat melakukan koneksi dengan broker karena menggunakan format username dengan token JWT dan password yang akan diabaikan nilainya, namun karena isi dari payload token JWT merupakan data yang tidak terdaftar dalam database maka permintaan koneksi akan ditolak. Karena mekanisme autentikasi yang dirancang menggunakan token JWT untuk pemeriksaan username dan password, sehingga pemeriksaan username dan password akan didapatkan dari payload pada token JWT oleh auth-server, lalu akan diperiksa di database dan auth-server gagal menemukan data klien pada database yang mengakibatkan broker untuk menolak permintaan koneksi klien. Berdasarkan hasil ini diketahui, bahwa sistem berhasil memeriksa token JWT dengan data klien pada database dan menolak permintaan koneksi klien yang tidak terdaftar dalam sistem. Hasil yang didapatkan oleh broker pada skenario ini dama dengan yang dilakukan pada skenario PMO_202, karena mekanisme autentikasi yang dibuat pada auth-server tidak diubah.

(8	0	*any																
	File	<u>E</u> dit	<u>V</u> iew	<u>G</u> o	<u>C</u> apture	<u>A</u> nalyze	<u>S</u> tatist	ics 1	Felephon <u>y</u>	<u>W</u> ire	eless <u>T</u> ools	s <u>H</u> elp							
			6	9	0103 0109 0119	8	9	(۵	•			⊕ ⊖	Ξ					
	Ap	oply a	display	/ filter	<ctrl-,< td=""><td>></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>- Exp</td><td>ression</td></ctrl-,<>	>												- Exp	ression
Ν	о.	Т	ime		Source			Destir	nation		Protocol	Length	Info						
	~	3 6 4 6 5 6 6 7 7 7 8 7 10 7 11 7 12 7 13 7 14 7 15 7 16 7	95714 95717 95742 03406 03403 03403 05455 209591 22714 227043 27566 27566 27566 27566 27668	13535 74124 24169 1151 32830 55319 10944 1546 32400 57866 06915 39663 12558 50292	192.10 192.11 192.11 192.11 192.11 192.10 192.10 192.10 192.10 192.11 192.11 192.11 192.11	38.100.29 38.100.40 58.100.29 58.100.29 58.100.40 58.100.40 58.100.29 58.100.29 58.100.29 58.100.29 58.100.40 58.100.29 58.100.40 58.100.40 58.100.40 58.100.40 58.100.40 58.100.40 58.100.40 58.100.40 58.100.40 58.100.40 58.100.40 58.100.40		192.1 192.1 192.1 192.1 192.1 192.1 192.1 192.1 192.1 192.1 192.1 192.1 192.1 192.1	68.100.4 68.100.2 68.100.4 68.100.4 68.100.2 68.100.2 68.100.2 68.100.2 68.100.4 68.100.2 68.100.4 68.100.4 68.100.4 68.100.4 68.100.4 68.100.4 68.100.4 68.100.4 68.100.4 68.100.4 68.100.4 68.100.4 68.100.4 68.100.4 68.100.4 68.100.4 68.100.4 68.100.2 68.100.2 68.100.2 68.100.2 68.100.2 68.100.2 68.100.2 68.100.2 68.100.2	0 9 0 9 9 9 9 9 9 9 9 9 9	TCP TCP TLSv1.2 TCP TLSv1.2 TCP TLSv1.2 TCP TLSv1.2 TCP TLSv1.2 TCP TLSv1.2 TCP TLSv1.2	68 68 62 250 56 1128 62 323 56 62 56 2 56 2 56 2 101 56 2 107	49217 8883 → 49217 Client 8883 → Server 49217 Client 8883 → Change 8883 → Hello 8883 → Change	→ 8883 49217 → 8883 Hello 49217 Hello, → 8883 Key E: 49217 Cipher 49217 Request 49217 Cipher	[SYN] [SYN, [ACK] [ACK] (ACK] (ACK] (ACK] (ACK] (ACK] (ACK] (ACK] (ACK] (ACK] (ACK] (ACK) (ACK)	Seq=0 W ACK] Se Seq=1 A ificate, Seq=195 e Seq=107 Seq=107 lo Reque Seq=107 , Encryp	in=64240 q=0 Ack= ck=1 Wir ck=195 W Server Ack=107 3 Ack=40 3 Ack=40 st 3 Ack=51 ted Hand) Len=(:1 Win: =5255(/in=303 Hello 3 Win: 3 Win: 3 Win: 3 Win: 1 Shake	0 MSS=1 29200 8 Len=0 36 Len Done 524288 31360 31360 31360 Message
	Fr Li In Tr Se	17 ame 1 nux c terne ansmi cure TLSv C V L E	.29471 7: 253 cooked t Prot ssion Socket 1.2 Re ontent ersion ength: ncrypt	3446 byte captu cocol Contr s Lay cord Type : TLS 192 ed Ap	192.1 es on win re Version ol Proto /er Layer: A : Applic 1.2 (0x plicatio	<pre>58.100.29 re (2024 H 4, Src: : ocol, Src upplication ation Dat 0303) n Data: 0</pre>	Dits), 3 192.168 Port: 4 Dn Data a (23)	192.1 253 b .100. 49217 Prot	68.100.4 ytes cap 29, Dst: , Dst Po ocol: mq 0001307e1	0 tured 192.: rt: 8 tt fc105d	TLSv1.2 (2024 bit 168.100.40 883, Seq: aab55c1430	253 s) on : 513, Ad	Applic interfac ck: 112 494ef7.	ation [ce 0 4, Len:	Data : 197				

Gambar 6.14 Hasil Capture Wireshark PMO_302

Seperti yang terlihat pada gambar 6.14, bahwa hasil *sniffing interface* didapatkan tidak ditemukan protokol MQTT, hal ini karena pertukaran paket data dilakukan melalui TLS. Hasil *sniffing* tidak mendapatkan pesan *CONNECT* yang dikirimkan oleh klien ke broker, namun jika dilihat berdasarkan paket yang dikirim dari ip klien menuju ip broker dapat dilihat pertukaran paket *Application Data* setelah melakukan handshake dan pertukaran sertifikat CA dan *key*. Jika dilihat pada isi data dari paket *Application Data* tersebut dapat dilihat bahwa data yang dikirimkan dapat tidak dapat dibaca karena dalam bentuk enkripsi data. Meskipun terdapat informasi pada paket tersebut berisi *username* dengan token JWT yang *invalid* dan *password* yang digunakan secara asal nilainya, namun informasi tersebut terlindungi oleh enkripsi di TLS.

6.2.1.8 Klien Membangun Koneksi Dengan Broker Dengan Menggunakan Token JWT yang Valid. (Kode: PMO_303)

Pada skenario pengujian ini digunakan konfigurasi broker MQTT yang akan menerapkan Auth-Server menggunakan *backend* JWT dan menerapkan keamanan TLS. Prosedur pengujian pada skenario ini akan dimulai dengan melakukan *capture interfaces* di Wireshark, lalu dengan aplikasi MQTT-Spy klien akan mengirim pesan *CONNECT* ke broker dengan menyertakan token JWT yang *valid*, yaitu token JWT milik pengguna yang terdaftar dalam database pada *field username*, dan *field password* dapat dikosongkan atau dapat diisi dengan nilai sembarangan, kemudian akan digunakan *Certificate of Authority* (CA) yang dibuat menggunakan OpenSSL. Setelah itu akan dilihat status broker, selanjutnya akan dilihat hasil *capture interface* terkait setiap paket yang dikirimkan berdasarkan alamat ip klien yang dikirim menuju alamat ip milik broker MQTT untuk melihat hasil *sniffing* di TLS.

1509621175: New connection from 192.168.100.29 on port 8883.
1509621175: mosquitto_auth_unpwd_check(eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.
eyJ1c2VybmFtZSI6ImNsaWVudDEiLCJwdyI6TmNvbnRvaDEyIn0.043ve3RXXPuYm0QVnc4CF-6uxT-8
_TU2ij70cqTKNME)
1509621175: ** checking backend jwt
1509621175: url=http://localhost:8100/auth
1509621175: data=topic=&acc=-1&clientid=
1509621175: getuser(eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmFtZSI6ImN
saWVudDEiLCJwdyI6ImNvbnRvaDEyIn0.043ve3RXXPuYm0QVnc4CF-6uxT-8_TU2ij7OcqTKNME) AU
THENTICATED=1 by jwt
1509621175: New client connected from 192.168.100.29 as Client_Pengujian (c1, k6
0, ueyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmFtZSI6ImNsaWVudDEiLCJwdyI6ImN
vbnRvaDEyIn0.043ve3RXXPuYm0QVnc4CF-6uxT-8_TU2ij7OcqTKNME).
1509621175: Sending CONNACK to Client_Pengujian (0)

Gambar 6.15 Screenshot Broker PMO_303

Pada gambar 6.9, dapat dilihat bahwa broker berhasil menerima permintaan koneksi dari klien dan mengirimkan pesan CONNACK bernilai 0, yang menandakan permintaan koneksi diterima, dengan menggunakan konfigurasi ini klien dapat melakukan koneksi dengan broker karena menggunakan format *username* dengan token JWT dan *password* yang akan diabaikan nilainya. Karena mekanisme autentikasi yang dirancang menggunakan token JWT untuk pemeriksaan *username* dan *password*, sehingga pemeriksaan *username* dan *password* akan didapatkan dari *payload* pada token JWT oleh auth-server, lalu akan diperiksa di

database dan auth-server berhasil menemukan data klien pada database yang mengakibatkan broker menerima permintaan koneksi klien. Berdasarkan hasil ini diketahui, bahwa sistem berhasil memeriksa token JWT dengan data klien pada database dan menerima permintaan koneksi klien yang terdaftar dalam sistem. Hasil yang didapatkan oleh broker pada skenario ini dama dengan yang dilakukan pada skenario PMO_203, karena mekanisme autentikasi yang dibuat pada authserver tidak diubah.

8		*any																		
<u>F</u> ile	<u>E</u> dit	<u>V</u> iew	<u>G</u> o	<u>C</u> apture	<u>A</u> nalyze	<u>S</u> tatist	ics	Telephon	<u>y W</u> i	reless	Tools	s <u>H</u> elp								
		6	9	00000	🗙 🏹	9		۵	•	垫			⊕ ∈		3					
	Apply a	display	filter	<ctrl- <="" th=""><th>></th><th></th><th></th><th></th><th></th><th></th><th></th><th></th><th></th><th></th><th></th><th></th><th>-</th><th>•</th><th>Expres</th><th>sion</th></ctrl->	>												-	•	Expres	sion
No.	Т	ime		Source			Desti	ination		Pr	otocol	Length	Info							
-	6 2 7 2 8 2 9 2 10 2 11 2 12 2 13 2 14 2 15 2 16 2 17 2	2.52616 2.52847 2.52817 2.52820 2.52873 2.53548 2.57844 2.57912 2.57914 2.57914 2.58197 2.58720 2.58720	5696 1674 0339 5490 8492 0247 7546 2260 2732 8258 8954 4479	192.16 192.16 192.16 192.16 192.16 192.16 192.16 192.16 192.16 192.16 192.16 192.16	88.100.40 88.100.29 88.100.40 88.100.40 88.100.40 88.100.40 88.100.29 88.100.40 88.100.40 88.100.40 88.100.40 88.100.40 88.100.40 88.100.40 88.100.40 88.100.40 88.100.40 88.100.40 88.100.40 88.100.40 88.100.40 9.1		192. 192. 192. 192. 192. 192. 192. 192.	168.100. 168.100. 168.100. 168.100. 168.100. 168.100. 168.100. 168.100. 168.100. 168.100. 168.100. 168.100. 168.100.	29 40 29 29 40 29 40 29 40 29 29 29 40	TC TC TL TC TL TC TL TC TL TC TL TC TC	P Sv1.2 Sv1.2 Sv1.2 Sv1.2 Sv1.2 P Sv1.2 P Sv1.2 Sv1.2 Sv1.2 Sv1.2	68 62 250 56 1128 323 56 107 56 107 56 107 264 76	8883 49244 Clien 8883 Serve Clien 8883 Chang 8883 Chang Appli 44668	→ 49244 → 8883 t Hello → 49244 r Hello t Key E → 49244 e Ciphe → 49244 e Ciphe Cation → 8100	[SYN, [ACK] [ACK] , Cert xchang [ACK] r Spec [ACK] r Spec Data [SYN]	ACK] S Seq=1 Seq=1 ificate e Seq=10 , Hello Seq=10 , Encry Seq=0	Geq=0 ACC Ack=1 W Ack=195 Ack=195 Ack=195 73 Ack=1 73 Ack=1 73 Ack=1 74 Ha Win=436	k=1 in=5 Win r He 462 t, H 513 ndsh 90 L	Win=29 25568 =30336 110 Do Win=31 ello R Win=31 ake Me en=0 M	200 Len=0 Len ne 360 eque 360 ssage SS=6
▶ F ▶ L ▶ 1 ▶ 1	rame 1 inux c nterne ransmi ecure TLSv C C U	6: 264 cooked t Prot ssion Socket 1.2 Re ontent ersion ength:	byte captu ocol Contr s Lay cord Type : TLS 203 ed App	version version col Proto ver Layer: A 1.2 (0x) plicatio	4, Src: 3 icol, Src: 3 pplication ation Dat 0303) n Data: 0	Dits), 192.168 Port: Dn Data a (23)	264 .100 4924 Prof	bytes ca .29, Dst 4, Dst P tocol: m	pture : 192 ort: 4 qtt 2fd69f	d (211 .168.1 8883,	2 bit: .00.40 Seq: !	s) on : 513, A	interf	24, Len	: 208					

Gambar 6.16 Hasil Capture Wireshark PMO_303

Seperti yang terlihat pada gambar 6.16, bahwa hasil *sniffing interface* didapatkan tidak ditemukan protokol MQTT, hal ini karena pertukaran paket data dilakukan melalui TLS. Hasil *sniffing* tidak mendapatkan pesan *CONNECT* yang dikirimkan oleh klien ke broker, namun jika dilihat berdasarkan paket yang dikirim dari ip klien menuju ip broker dapat dilihat pertukaran paket *Application Data* setelah melakukan handshake dan pertukaran sertifikat CA dan *key*. Jika dilihat pada isi data dari paket *Application Data* tersebut dapat dilihat bahwa data yang dikirimkan dapat tidak dapat dibaca karena dalam bentuk enkripsi data. Meskipun terdapat informasi pada paket tersebut berisi *username* dengan token JWT yang *valid* dan *password* yang digunakan secara asal nilainya, namun informasi tersebut terlindungi oleh enkripsi di TLS.

6.2.2 Pengujian Mekanisme Otorisasi Sistem Melalui Publish

Pengujian ini dilakukan untuk mengetahui hasil penerapan mekanisme otorisasi pada sistem berbasis protokol MQTT pada saat klien melakukan *publish* ke broker. Pengujian ini akan membandingkan bagaimana perbedaan keamanan pada ketiga konfigurasi broker MQTT yang sudah dijabarkan, ketiga konfigurasi broker akan memiliki beberapa skenario yang akan dilakukan. Pengujian akan dilakukan dengan cara klien MQTT yang sudah terkoneksi dengan broker dan berhasil terautentikasi, akan mengirimkan pesan *Publish* ke broker MQTT, kemudian pada tiap skenario akan digunakan parameter yang berbeda terkait topik yang akan digunakan, apakah dengan sesuai dengan *topic tree*, atau tidak dan sesuai dengan hak akses atau tidak. Pada pengujian ini, pertama-tama Wireshark akan dijalankan dan melakukan *capture* kemudian klien MQTT akan mengirimkan pesan *publish* ke broker MQTT dengan topik yang disesuaikan pada tiap skenario. Pesan *publish* akan dikirimkan menggunakan aplikasi dashboard MQTT-spy dengan menggunakan data klien yang terdaftar dalam database, setelah dilakukan pengiriman pesan *publish* dari klien selanjutnya akan dilihat hasil *capture* dari Wireshark dan status dari mosquitto-broker dan juga auth-server jika digunakan.

6.2.2.1 Klien Melakukan *Publish* Pada Topik Yang Tidak Ada Dalam *Topic Tree* Yang Dibuat. (Kode: PMOP_101)

Pada skenario pengujian ini digunakan konfigurasi broker MQTT yang tidak menerapkan Auth-Server dan tidak menerapkan TLS. Prosedur pengujian pada skenario ini akan dimulai dengan melakukan *capture interfaces* di Wireshark, lalu dengan aplikasi MQTT-Spy klien akan mengirim pesan *PUBLISH* pada topik yang tidak terdapat dalam *topic tree* ke broker, setelah itu akan dilihat status broker, selanjutnya akan dilakukan *filter* pada hasil *capture interface* menggunakan *field* MQTT yang akan menyeleksi setiap paket yang dikirimkan menggunakan protokol MQTT untuk melihat hasil *sniffing*.

```
Gambar 6.17 Screenshot Broker PMOP_101
```

Pada gambar 6.17, dapat dilihat bahwa broker menerima pesan *PUBLISH* dari klien, dengan menggunakan konfigurasi ini klien dapat mengirimkan pesan *publish* pada topik manapun baik yang dirancang pada *topic tree* maupun diluar dari rancangan, sehingga dibutuhkan sebuah mekanisme otorisasi untuk mengatur hak akses pengguna dan topik yang dapat diakses.

8	•	•	*any														
Fi	le	<u>E</u> dit	<u>V</u> iew	Go	<u>C</u> apture	<u>A</u> nalyze	<u>S</u> tatis	tics Te	ephony	<u>W</u> ire	less	<u>T</u> ools	He	lp			
			6	9	0251 0210 0211	8	9	<pre></pre>) 🖄	^	Ł			Ð	Q	€	
	Ap	ply a	display	filter	· <ctrl- <="" th=""><th>></th><th></th><th></th><th></th><th></th><th></th><th></th><th></th><th></th><th>•</th><th>Expr</th><th>essio</th></ctrl->	>									•	Expr	essio
No.		Time	Sourc	e		Destinat	ion	P	rotocol	Length	Info						
	23	21	192.3	168.1	.00.29	192.168	100.4	0 M	QTT	111	Conne	ect C	omma	nd			
	24	21	192.3	168.1	.00.40	192.168	.100.2	9 T	ČР	56	1883	→ 61	866	[ACK]	Seq=1	1 Ack	=56
	25	21	192.3	168.1	.00.40	192.168	.100.2	9 M	QTT	60	Conne	ect A	ck				
	26	21	192.3	168.1	.00.29	192.168	100.4	0 T	CP	62	6186	6 → 1	883	[ACK]	Seq=	56 Ac	k=5 \
	27	26	PcsC	ompu_	7d:88:2f			A	RP	44	Who	has 1	92.1	68.100	9.29?	Tell	192
	28	26	Lite	onTe_	_c9:03:8f			A	RP	62	192.3	168.1	00.2	9 is a	at 5c	:93:a	2:c9
	29	31	192.3	168.1	.00.29	192.168	.100.2	55 D	B-LSP	177	Drop	box L	AN s	ync D:	iscove	ery P	roto
	30	50	192.3	168.1	.00.29	192.168	.100.4	0 M	QTT	112	Pub1:	ish M	essa	ge			
L	31	50	192.3	168.1	.00.40	192.168	.100.2	9 T	CP	56	1883	→ 61	866	[ACK]	Seq=	Ack	=112
	32	55	Lite	onTe_	_c9:03:8f			A	RP	62	Who	has 1	92.1	68.100	9.40?	Tell	192
	33	55	PCSC	ompu_	/d:88:2f			A	RP	44	192.3	168.1	90.4	0 1S 8	at 08	:00:2	/:/d
•	Fra	ume 3	9: 112	byte	es on wir	e (896 b:	its), 1	12 byte	s captı	ured (896 b	its)	on i	nterf	ace 0		
•	Lin	IUX C	ooked	capti	ure												
•	Int	erne	t Prot	ocol	Version	4, Src: :	192.168	3.100.29	, Dst:	192.1	68.10	0.40					
	Tra	insmi	ssion	Conti	rol Proto	col, Src	Port:	61866,	Dst Por	rt: 18	83, S	eq: 5	6, A	ck: 5	, Len	: 56	
•	MQ	Tete	netry	Iran	sport Pro	tocol											
	•	Publi	LSN Me	ssage													
		00	11 000	00 =	Header F.	Lags: 0x3	i⊎ (Pub	iisn Me	ssage)								
		MS To	g Len	: 54	(-										
		10	pic: 1	сорік	/pengujia	an/apaSaj	a a dili	ur d m									
		Me	ssage	: 1N1	. adalah j	besan yan	ig aiki	rım									

Gambar 6.18 Hasil Capture Wireshark PMOP_101

Seperti yang terlihat pada gambar 6.18, bahwa hasil *sniffing* protokol MQTT mendapatkan pesan *PUBLISH* yang dikirimkan oleh klien ke broker. Jika dilihat pada isi data dari paket *publish* tersebut dapat dilihat bahwa data yang dikirimkan dapat dibaca dalam bentuk *plain text*, kemudian jika dilihat isi dari pesan *publish* tersebut didalamnya terdapat *field Topic* yang memiliki nilai "topik/pengujian/apaSaja" dengan *filed Message* yang memiliki nilai "ini adalah pesan yang dikirim".

6.2.2.2 Klien Melakukan *Publish* Pada Topik Yang Ada Dalam *Topic Tree* Yang Dibuat. (Kode: PMOP_102)

Pada skenario pengujian ini digunakan konfigurasi broker MQTT yang tidak menerapkan Auth-Server dan tidak menerapkan TLS. Prosedur pengujian pada skenario ini akan dimulai dengan melakukan *capture interfaces* di Wireshark, lalu dengan aplikasi MQTT-Spy klien akan mengirim pesan *PUBLISH* pada topik yang terdapat dalam *topic tree* ke broker, setelah itu akan dilihat status broker, selanjutnya akan dilakukan *filter* pada hasil *capture interface* menggunakan *field* MQTT yang akan menyeleksi setiap paket yang dikirimkan menggunakan protokol MQTT untuk melihat hasil *sniffing*.

```
Gambar 6.19 Screenshot Broker PMOP_102
```

Pada gambar 6.19, dapat dilihat bahwa broker menerima pesan PUBLISH dari klien, dengan menggunakan konfigurasi ini klien dapat mengirimkan pesan publish pada topik manapun baik yang dirancang pada *topic tree* maupun diluar dari rancangan, sehingga dibutuhkan sebuah mekanisme otorisasi untuk mengatur hak akses pengguna dan topik yang dapat diakses.

8		any			
<u>F</u> ile	<u>E</u> dit	<u>V</u> iew <u>G</u> o <u>C</u> aptur	e <u>A</u> nalyze <u>S</u> tatistics	Telephony	<u>y W</u> ireless <u>T</u> ools <u>H</u> elp
		6 💿 🚞 🚺) 🖹 🎑 🤇 ⇐) 🔿 🔮	🛃 📃 🗐 🍳 🍳 💵
4	Apply a	display filter <ctr< td=""><td>-/></td><td></td><td>Expressi</td></ctr<>	-/>		Expressi
No.	Time	Source	Destination	Protocol	Length Info
E.	1 0.0	192.168.100.29	192.168.100.40	MQTT	127 Publish Message
	2 0.0	192.168.100.40	192.168.100.29	TCP	56 1883 → 61920 [ACK] Seq=1 Ack=72
	3 4.9	LiteonTe_c9:03:8	f	ARP	62 Who has 192.168.100.40? Tell 192
	4 4.9	PcsCompu 7d:88:2	f	ARP	44 192.168.100.40 is at 08:00:27:70
	5 5.1	PcsCompu 7d:88:2	f	ARP	44 Who has 192.168.100.29? Tell 192
	6 5.1	LiteonTe_c9:03:8	f	ARP	62 192.168.100.29 is at 5c:93:a2:c
	7 7.3	192.168.100.40	192.168.100.29	TCP	56 1883 → 61920 [FIN, ACK] Seg=1 A
	8 7.3	192.168.100.29	192.168.100.40	TCP	62 61920 → 1883 [ACK] Seg=72 Ack=2
	9 7.3	192.168.100.29	192.168.100.40	TCP	62 61920 → 1883 [FIN, ACK] Seg=72 /
L 1	.0 7.3	192.168.100.40	192.168.100.29	TCP	56 1883 → 61920 [ACK] Seq=2 Ack=73

Frame 1: 127 bytes on wire (1016 bits), 127 bytes captured (1016 bits) on interface 0
 Linux cooked capture

Transmission Control Protocol, Src Port: 61920, Dst Port: 1883, Seq: 1, Ack: 1, Len: 71

- 0011 0000 = Header Flags: 0x30 (Publish Message)
- Msg Len: 69

Topic: nodeMCU/LED/red/toggle Message: ini pesan pada topik sesuai dengan topic tree

Gambar 6.20 Hasil Capture Wireshark PMOP_102

Internet Protocol Version 4, Src: 192.168.100.29, Dst: 192.168.100.40

Publish Message

Seperti yang terlihat pada gambar 6.20, bahwa hasil *sniffing* protokol MQTT mendapatkan pesan *PUBLISH* yang dikirimkan oleh klien ke broker. Jika dilihat pada isi data dari paket *publish* tersebut dapat dilihat bahwa data yang dikirimkan dapat dibaca dalam bentuk *plain text*, kemudian jika dilihat isi dari pesan *publish* tersebut didalamnya terdapat *field Topic* yang memiliki nilai "nodeMCU/LED/red/toggle" dengan *filed Message* yang memiliki nilai "ini pesan pada topik sesuai dengan *topic tree*". Dari hasil yang didapatkan pada skenario ini adalah sama dengan yang didapatkan pada skenario PMOP_101.

6.2.2.3 Klien Melakukan *Publish* Pada Topik Diluar *Topic Tree* Yang Dibuat. (Kode: PMOP_201)

Pada skenario pengujian ini digunakan konfigurasi broker MQTT yang akan menerapkan Auth-Server menggunakan *backend* JWT namun tidak menerapkan TLS. Prosedur pengujian pada skenario ini akan dimulai dengan melakukan *capture interfaces* di Wireshark, lalu dengan aplikasi MQTT-Spy klien akan mengirim pesan *PUBLISH* pada topik yang tidak terdapat dalam *topic tree* ke broker, setelah itu akan dilihat status broker, selanjutnya akan dilakukan *filter* pada hasil *capture interface* menggunakan *field* MQTT yang akan menyeleksi setiap paket yang dikirimkan menggunakan protokol MQTT untuk melihat hasil *sniffing*.

1509782495: url=http://localhost:8100/acl
1509782495: data=topic=topik%2Fpengujian%2FapaSaja&a
cc=2&clientid=Klien_Pengujian
1509782495: aclcheck(eyJhbGciOiJIUzI1NiIsInR5cCI6Ikp
XVCJ9.eyJ1c2VybmFtZSI6InNpTWVyYWgiLCJwdyI6InJlZDEyMzQ1In
0.vvcTUByumkomj-oxQpqrGVkjK89zncElr4kFKxgkIQI, topik/pen
gujian/apaSaja, 2) AUTHORIZED=0 by (null)
1509782495: Cached [A2111F36FDE7DCBF01FDA55E93218B
F7D7A0F7C0] for (Klien_Pengujian,eyJhbGci0iJIUzI1NiIsInR
5cCI6IkpXVCJ9.eyJ1c2VybmFtZSI6InNpTWVyYWgiLCJwdyI6InJlZD
EyMzQ1In0.vvcTUByumkomj-oxQpqrGVkjK89zncElr4kFKxgkIQI,2)
1509782495: Denied PUBLISH from Klien_Pengujian (d0, q0,
r0, m0, 'topik/pengujian/apaSaja', (31 bytes))

Gambar 6.21 Screenshot Broker PMOP_201

Pada gambar 6.21, dapat dilihat bahwa broker menerima sebuah pesan *PUBLISH* kemudian mengirimkan pesan tersebut ke url milik auth-server yang menangani mekanisme otorisasi yaitu "http://localhost:8100/acl", dengan data *topic* yang bernilai "topik/pengujian/apaSaja", acc yang bernilai "2" yang manyatakan bahwa ini merupakan pesan *publish* serta clientid milik klien. Setelah itu diterima respon dari auth-server terkait pesan tersebut, dan dinyatakan bahwa pengguna tersebut tidak terotorisasi untuk melakukan *publish* di topik tersebut, yang dinyatakan dengan "*AUTHORIZED*=0". Lalu broker melakukan "*Denied PUBLISH*" pesan tersebut. Konfigurasi dengan auth-server ini berhasil menolak pesan *publish* yang dikirimkan pada topik yang tidak terdapat pada *topic tree* yang diperiksa pada ACL di database. Dikarenakan setiap topik yang dapat diakses sudah dirancang sebelumnya sehingga pengguna tidak dapat mengakses topik diluar dari perancancangan topik tersebut.

	8	0	Captur	ring f	rom any											
	<u>F</u> ile	<u>E</u> dit	<u>V</u> iew	<u>G</u> o	<u>C</u> apture	<u>A</u> naly	ze	<u>S</u> tatistics	Telephon	<u>y W</u> ire	less	<u>T</u> ools	<u>H</u> elp			
			ی ا		0101 0110 0113	X	C	۹ 🔶	• 🔿 🙅		Ł		Ð	Q	€) (
	A	oply a	display	filter	<ctrl- <="" td=""><td>></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>Exp</td><td>ressic</td></ctrl->	>									Exp	ressic
N	lo.	Time	Sourc	e		Desti	natio	n	Protoco	Length	Info					
	- 1 2	0.0.	<u>192_1</u> Time (f	ormat	as specif	192. ied) 2.	168. 168.:	100.40 100.29	MQTT TCP	114 56	Pub 188	lish Mes 3 → 6192	sage 8 [ACK]	Seq=1	1 Ack	(=59
	3	0.0	127.0	0.0.1 0.0.1		127. 127.	0.0.:	1	TCP TCP	76 76	393 810	56 → 810 9 → 3935	0 [SYN] 6 [SYN,	Seq=(ACK]	9 Wir Seq=	1=436 =0 Ac
	6	0.0	127.0	0.0.1		127.	0.0.	1	HTTP	68 376	393	56 → 816 T /super	USER HT	Seq=1	1 ACH 1 (a	(=1 W appli
	8	0.0	127.0	0.0.1		127.	0.0.	1	TCP	92	[TCI	9 → 3930 P segmen	t of a	reasse Sog=2	emble	ed PD
	10	0.1	127.0	0.0.1		127.	0.0.1	1	TCP	105	[TCI 393	P segmer 56 → 810	t of a	reasse Seg=3	emble 309 #	ed PD
	Fr Li In Tr	ame 1 nux co ternet ansmis	: 114 boked t Protession	bytes captu ocol Contr	on wire re Version ol Proto	(912 4, Sro	bits : 19 Src P), 114 b 2.168.10 ort: 619	ytes capt 0.29, Dst 28, Dst P	ured (9 : 192.1 ort: 18	012 b 68.1 83,	its) on 00.40 Seq: 1,	interfa	ace 0	58	
	MQ	Teler	netry i	Trans	port Pro	tocol										
	•	Publi ▶ 00 Ms To Me	sn Mes 11 000 g Len: pic: t ssage:	ssage)0 = 56 :opik pub:	Header Fi /pengujia Lish pesa	lags: an/apa an dil	0x30 Saja uar	(Publis topic tr	h Message) ee)						

Gambar 6.22 Hasil Capture Wireshark PMOP_201

Seperti yang terlihat pada gambar 6.22, bahwa hasil *sniffing* protokol MQTT mendapatkan pesan *PUBLISH* yang dikirimkan oleh klien ke broker. Jika dilihat pada isi data dari paket *publish* tersebut dapat dilihat bahwa data yang dikirimkan dapat dibaca dalam bentuk *plain text*, kemudian jika dilihat isi dari pesan *publish* tersebut didalamnya terdapat *field Topic* yang memiliki nilai "topik/pengujian/apaSaja" dengan *filed Message* yang memiliki nilai "*publish* pesan diluar *topic tree*".

6.2.2.4 Klien Melakukan *Publish* Pada Topik Yang Tidak Sesuai Dengan Hak Akses Di *Topic Tree* Yang Dibuat. (Kode: PMOP_202)

Pada skenario pengujian ini digunakan konfigurasi broker MQTT yang akan menerapkan Auth-Server menggunakan *backend* JWT namun tidak menerapkan TLS. Prosedur pengujian pada skenario ini akan dimulai dengan melakukan *capture interfaces* di Wireshark, lalu dengan aplikasi MQTT-Spy klien akan mengirim pesan *PUBLISH* pada topik yang terdapat dalam *topic tree* namun diluar dari hak akses milik pengguna tersebut ke broker, setelah itu akan dilihat status broker, selanjutnya akan dilakukan *filter* pada hasil *capture interface* menggunakan *field* MQTT yang akan menyeleksi setiap paket yang dikirimkan menggunakan protokol MQTT untuk melihat hasil *sniffing*. Pada pengujian ini digunakan pengguna yang terdaftar dalam sistem yaitu siMerah yang memiliki peran sebagai red_user.

Pada gambar 6.23, dapat dilihat bahwa broker menerima sebuah pesan *PUBLISH* kemudian mengirimkan pesan tersebut ke url milik auth-server yang menangani mekanisme otorisasi yaitu "http://localhost:8100/acl", dengan data *topic* yang bernilai "nodeMCU/DHT/temperature/status", acc yang bernilai "2" yang manyatakan bahwa ini merupakan pesan *publish* serta clientid milik klien. Setelah itu diterima respon dari auth-server terkait pesan tersebut, dan

dinyatakan bahwa pengguna tersebut tidak terotorisasi untuk melakukan *publish* di topik tersebut, yang dinyatakan dengan "*AUTHORIZED*=0". Lalu broker melakukan "*Denied PUBLISH*" pesan tersebut. Konfigurasi dengan auth-server ini berhasil menolak pesan *publish* yang dikirimkan pada topik yang terdapat pada *topic tree* namun diluar dari hak akses milik pengguna yang diperiksa pada ACL di database. Dikarenakan pengguna memiliki peran tertentu, dan pada peran tersebut telah dirancang tiap topik yang dapat diakses, maka broker berhasil menolak *publish* diluar hak akses milik pengguna tersebut.

1509782887: url=http://localhost:8100/acl
1509782887: data=topic=nodeMCU%2FDHT%2Ftemperature%2
Fstatus&acc=2&clientid=Klien_Pengujian
1509782887: aclcheck(eyJhbGciOiJIUzI1NiIsInR5cCI6Ikp
XVCJ9.eyJ1c2VybmFtZSI6InNpTWVyYWgiLCJwdyI6InJlZDEyMzQ1In
0.vvcTUByumkomj-oxQpqrGVkjK89zncElr4kFKxgkIQI, nodeMCU/D
HT/temperature/status, 2) AUTHORIZED=0 by (null)
1509782887: Cached [19F02969E6AFD90E4776FBF107F296
8F4F9DFA5B] for (Klien_Pengujian,eyJhbGci0iJIUzI1NiIsInR
5cCI6IkpXVCJ9.eyJ1c2VybmFtZSI6InNpTWVyYWgiLCJwdyI6InJlZD
EyMzQ1In0.vvcTUByumkomj-oxQpqrGVkjK89zncElr4kFKxgkIQI,2)
1509782887: Denied PUBLISH from Klien_Pengujian (d0, q0,
r0, m0, 'nodeMCU/DHT/temperature/status', (30 bytes

Gambar 6.23 Screenshot Broker PMOP_202

Seperti yang terlihat pada gambar 6.24, bahwa hasil *sniffing* protokol MQTT mendapatkan pesan *PUBLISH* yang dikirimkan oleh klien ke broker. Jika dilihat pada isi data dari paket *publish* tersebut dapat dilihat bahwa data yang dikirimkan dapat dibaca dalam bentuk *plain text*, kemudian jika dilihat isi dari pesan *publish* tersebut didalamnya terdapat *field Topic* yang memiliki nilai "nodeMCU/DHT/temperature/status" dengan *filed Message* yang memiliki nilai "*publish* pesan diluar hak akses".

8		*any															
<u>F</u> ile	<u>E</u> dit	<u>V</u> iew	<u>G</u> o <u>C</u>	apture	<u>A</u> nalyze	e <u>S</u> tat	istics	Teleph	ion <u>y</u>	<u>W</u> ir	eless	Too	ls <u>I</u>	lelp			
		6		0101 0110 0111	8			i)	1	⊵			9	Q	Q	•
m	qtt													×	•	Exp	ressio
No.	Time	Source	e		Destina	ation		Proto	col	Lengt	th Info)					
L C S	3 6.9	192.1	68.100	0.29	192.10	68.100.	40	MQTT		12	0 Pub	lish	Mess	sage			
 Fr Li In Tr 	 Frame 3: 120 bytes on wire (960 bits), 120 bytes captured (960 bits) on interface 0 Linux cooked capture Internet Protocol Version 4, Src: 192.168.100.29, Dst: 192.168.100.40 Transmission Control Protocol, Src Port: 61941, Dst Port: 1883, Seq: 1, Ack: 1, Len: 64 																
₩Q ₩Q	Tele Publ: • 00 Ms To Me	metry T ish Mes 011 000 sg Len: opic: n essage:	ranspo sage 0 = He 62 odeMCU publi	ort Pro Bader FJ J/DHT/te ish pesa	tocol Lags: 0> emperatu an dilua	(30 (Pu Ire/sta ar hak	ıblish itus akses	Messa	ge)								

Gambar 6.24 Hasil Capture Wireshark PMOP_202

6.2.2.5 Klien Melakukan *Publish* Pada Topik Yang Sesuai Dengan Hak Akses Di *Topic Tree* Yang Dibuat. (Kode: PMOP_203)

Pada skenario pengujian ini digunakan konfigurasi broker MQTT yang akan menerapkan Auth-Server menggunakan *backend* JWT namun tidak menerapkan TLS. Prosedur pengujian pada skenario ini akan dimulai dengan melakukan *capture interfaces* di Wireshark, lalu dengan aplikasi MQTT-Spy klien akan mengirim pesan *PUBLISH* pada topik yang terdapat dalam *topic tree* dan sesuai dengan hak akses milik pengguna tersebut ke broker, setelah itu akan dilihat status broker, selanjutnya akan dilakukan *filter* pada hasil *capture interface* menggunakan *field* MQTT yang akan menyeleksi setiap paket yang dikirimkan menggunakan protokol MQTT untuk melihat hasil *sniffing*. Pada pengujian ini digunakan pengguna yang terdaftar dalam sistem yaitu siMerah yang memiliki peran sebagai red_user.

Pada gambar 6.25, dapat dilihat bahwa broker menerima sebuah pesan *PUBLISH* kemudian mengirimkan pesan tersebut ke url milik auth-server yang menangani mekanisme otorisasi yaitu "http://localhost:8100/acl", dengan data *topic* yang bernilai "nodeMCU/LED/red/toggle", acc yang bernilai "2" yang manyatakan bahwa ini merupakan pesan *publish* serta clientid milik klien. Setelah itu diterima respon dari auth-server terkait pesan tersebut, dan dinyatakan bahwa pengguna tersebut terotorisasi untuk melakukan *publish* di topik tersebut, yang dinyatakan dengan "*AUTHORIZED*=1". Lalu broker melakukan "*Received PUBLISH*" pesan tersebut, yang menandakan bahwa pesan publish berhasil diterima dan akan diteruskan pada *subscriber* pada topik tersebut. Konfigurasi dengan auth-server ini berhasil menerima pesan *publish* yang dikirimkan pada topik yang terdapat pada *topic tree* dan sesuai dengan hak akses milik pengguna yang diperiksa pada ACL di database.

1509783397: url=http://localhost:8100/acl
1509783397: data=topic=nodeMCU%2FLED%2Fred%2Ftoggle&
acc=2&clientid=Klien_Pengujian
1509783397: aclcheck(eyJhbGciOiJIUzI1NiIsInR5cCI6Ikp
XVCJ9.eyJ1c2VybmFtZSI6InNpTWVyYWgiLCJwdyI6InJlZDEyMzQ1In
0.vvcTUByumkomj-oxQpqrGVkjK89zncElr4kFKxgkIQI, nodeMCU/L
ED/red/toggle, 2) trying to acl with jwt
1509783397: aclcheck(eyJhbGciOiJIUzI1NiIsInR5cCI6Ikp
XVCJ9.eyJ1c2VybmFtZSI6InNpTWVyYWgiLCJwdyI6InJlZDEyMzQ1In
0.vvcTUByumkomj-oxQpqrGVkjK89zncElr4kFKxgkIQI, nodeMCU/L
ED/red/toggle, 2) AUTHORIZED=1 by jwt
1509783397: Cached [5B8F665F3B22379223AEA0EBDFBA01
83CADE5F86] for (Klien_Pengujian,eyJhbGci0iJIUzI1NiIsInR
5cCI6IkpXVCJ9.eyJ1c2VybmFtZSI6InNpTWVyYWgiLCJwdyI6InJlZD
EyMzQ1In0.vvcTUByumkomj-oxQpqrGVkjK89zncElr4kFKxgkIQI,2)
1509783397: Received PUBLISH from Klien_Pengujian (d0, q
0, r0, m0, 'nodeMCU/LED/red/toggle', (30 bytes))

Gambar 6.25 Screenshot Broker PMOP_203

Seperti yang terlihat pada gambar 6.24, bahwa hasil *sniffing* protokol MQTT mendapatkan pesan *PUBLISH* yang dikirimkan oleh klien ke broker. Jika dilihat pada isi data dari paket *publish* tersebut dapat dilihat bahwa data yang dikirimkan dapat dibaca dalam bentuk *plain text*, kemudian jika dilihat isi dari pesan *publish*

tersebut didalamnya terdapat *field Topic* yang memiliki nilai "nodeMCU/LED/red/toggle" dengan *filed Message* yang memiliki nilai "*publish* pesan sesuai hak akses". Meskipun berdasarkan keberhasilan mekanisme otorisasi pada pengujian PMOP_201, PMOP_202 dan PMOP_203, namun hasil *sniffing* berhasil mendapatkan topik dan pesan yang dipublish oleh klien pada sistem.



Gambar 6.26 Hasil Capture Wireshark PMOP_203

6.2.2.6 Klien Melakukan *Publish* Pada Topik Diluar *Topic Tree* Yang Dibuat. (Kode: PMOP_301)

Pada skenario pengujian ini digunakan konfigurasi broker MQTT yang akan menerapkan Auth-Server menggunakan *backend* JWT dan menerapkan keamanan TLS. Prosedur pengujian pada skenario ini akan dimulai dengan melakukan *capture interfaces* di Wireshark lalu dengan aplikasi MQTT-Spy klien akan mengirim pesan *PUBLISH* pada topik yang tidak terdapat dalam *topic tree* ke broker. Setelah itu akan dilihat status broker, selanjutnya akan dilihat hasil *capture interface* terkait setiap paket yang dikirimkan berdasarkan alamat ip klien yang dikirim menuju alamat ip milik broker MQTT untuk melihat hasil *sniffing* di TLS.

Pada gambar 6.27, dapat dilihat bahwa broker menerima sebuah pesan *PUBLISH* kemudian mengirimkan pesan tersebut ke url milik auth-server yang menangani mekanisme otorisasi yaitu "http://localhost:8100/acl", dengan data *topic* yang bernilai "topik/pengujian/apaSaja", acc yang bernilai "2" yang manyatakan bahwa ini merupakan pesan *publish* serta clientid milik klien. Setelah itu diterima respon dari auth-server terkait pesan tersebut, dan dinyatakan bahwa pengguna tersebut tidak terotorisasi untuk melakukan *publish* di topik tersebut, yang dinyatakan dengan "*AUTHORIZED*=0". Lalu broker melakukan "*Denied PUBLISH*" pesan tersebut. Konfigurasi dengan auth-server ini berhasil menolak pesan *publish* yang dikirimkan pada topik yang tidak terdapat pada *topic tree* yang diperiksa pada ACL di database. Dikarenakan setiap topik yang dapat diakses sudah dirancang sebelumnya sehingga pengguna tidak dapat mengakses topik diluar dari perancancangan topik tersebut. Hasil yang didapatkan oleh broker pada

skenario ini dama dengan yang dilakukan pada skenario PMOP_201, karena mekanisme autentikasi yang dibuat pada auth-server tidak diubah.

1509783803: url=http://localhost:8100/acl 1509783803: data=topic=topik%2Fpengujian%2FapaSaja&a cc=2&clientid=Klien_Pengujian 1509783803: aclcheck(eyJhbGci0iJIUzI1NiIsInR5cCI6Ikp
1509783803: data=topic=topik%2Fpengujian%2FapaSaja&a cc=2&clientid=Klien_Pengujian 1509783803: aclcheck(eyJhbGci0iJIUzI1NiIsInR5cCI6Ikp
cc=2&clientid=Klien_Pengujian 1509783803: aclcheck(eyJhbGci0iJIUzI1NiIsInR5cCI6Ikp
1509783803: aclcheck(eyJhbGciOiJIUzI1NiIsInR5cCI6Ikp
XVCJ9.eyJ1c2VybmFtZSI6InNpTWVyYWgiLCJwdyI6InJlZDEyMzQ1In
0.vvcTUByumkomj-oxQpqrGVkjK89zncElr4kFKxgkIQI, topik/pen
gujian/apaSaja, 2) AUTHORIZED=0 by (null)
1509783803: Cached [A2111F36FDE7DCBF01FDA55E93218B
F7D7A0F7C0] for (Klien Pengujian.eyJhbGciOiJIUzI1NiIsInR
5cCI6IkpXVCJ9.eyJ1c2VybmFtZSI6InNpTWVyYWgiLCJwdyI6InJlZD
EyMz01In0.vvcTUByumkomj-ox0pgrGVkjK89zncElr4kFKxgkIQI,2)
1509783803: Denied PUBLISH from Klien Pengujian (d0, q0,
r0, m0, 'topik/pengujian/apaSaja', (31 bytes))

Gambar 6.27 Screenshot Broker PMOP_301

Seperti yang terlihat pada gambar 6.28, bahwa hasil *sniffing interface* didapatkan tidak ditemukan protokol MQTT, hal ini karena pertukaran paket data dilakukan melalui TLS. Hasil *sniffing* tidak mendapatkan pesan *PUBLISH* yang dikirimkan oleh klien ke broker, namun jika dilihat berdasarkan paket yang dikirim dari ip klien menuju ip broker dapat dilihat pertukaran paket *Application Data*. Jika dilihat pada isi data dari paket *Application Data* tersebut dapat dilihat bahwa data yang dikirimkan dapat tidak dapat dibaca karena dalam bentuk enkripsi data.

8	•	any										
<u>F</u> ile	<u>E</u> dit	<u>V</u> iew <u>G</u> o	<u>C</u> apture	<u>A</u> nalyze <u>S</u>	tatistics Te	elephon <u>y</u>	<u>W</u> irele	ss <u>T</u> ools	<u>H</u> elp			
			0101 0110 0111	8	۹ 🔶 🗖) 🖄	<u></u>	Ł 📃 ((Θ	€	•
	opply a c	display filter	r <ctrl- <="" th=""><th>></th><th></th><th></th><th></th><th></th><th></th><th></th><th>Expre</th><th>essio</th></ctrl->	>							Expre	essio
No.	Time	Source		Destination	F	Protocol	Length II	nfo				
 Fi L: Ti 	2 5.6 3 5.6 4 5.6 5 5.6 6 5.6 rame 2: inux co nternet ransmis	192.168.1 192.168.1 127.0.0.1 127.0.0.1 127.0.0.1 143 bytes oked capto Protocol ssion Conto	L00.29 L00.40 L L S on wire ure Version rol Proto	192.168.10 192.168.10 127.0.0.1 127.0.0.1 127.0.0.1 (1144 bits 4, Src: 192 col, Src Po	00.40 00.29), 143 byt .168.100.2 rt: 61956,	TLSv1.2 TCP TCP TCP TCP es captu 9, Dst: Dst Po	143 A 56 8 76 3 76 8 68 3 ured (11 192.168 rt: 888	pplicati 1883 → 61 19374 → 8 1100 → 39 19374 → 8 144 bits) 8.100.40 3, Seq: 1	on Data 956 [ACK 100 [SYN 374 [SYN 100 [ACK) on inte L, Ack: 1] Seq=] Seq= , ACK]] Seq= erface , Len:	1 Ack= 9 Win= Seq=0 1 Ack= 0 87	=88 =436 Ə Ac =1 W
V St	ecure S TLSv1 Con Ve Len End	ockets Lay .2 Record ntent Type rsion: TLS ngth: 82 crypted Ap	yer Layer: Ap e: Applica 5 1.2 (0x0 pplication	pplication ation Data 0303) n Data: 0000	Data Proto (23) 000000000000	col: mq1 002d8125	tt 5310ef2a	b628cd88	b0b5b77c	a2f7		

Gambar 6.28 Hasil Capture Wireshark PMOP_301

6.2.2.7 Klien Melakukan *Publish* Pada Topik Yang Tidak Sesuai Dengan Hak Akses Di *Topic Tree* Yang Dibuat. (Kode: PMOP_302)

Pada skenario pengujian ini digunakan konfigurasi broker MQTT yang akan menerapkan Auth-Server menggunakan *backend* JWT dan menerapkan keamanan TLS. Prosedur pengujian pada skenario ini akan dimulai dengan melakukan *capture interfaces* di Wireshark lalu dengan aplikasi MQTT-Spy klien akan mengirim pesan *PUBLISH* pada topik yang terdapat dalam *topic tree* namun diluar dari hak akses

milik pengguna tersebut ke broker, setelah itu akan dilihat status broker, selanjutnya akan dilakukan *filter* pada hasil *capture interface* menggunakan *field* MQTT yang akan menyeleksi setiap paket yang dikirimkan menggunakan protokol MQTT untuk melihat hasil *sniffing*. Pada pengujian ini digunakan pengguna yang terdaftar dalam sistem yaitu siMerah yang memiliki peran sebagai red_user.

Pada gambar 6.29, dapat dilihat bahwa broker menerima sebuah pesan PUBLISH kemudian mengirimkan pesan tersebut ke url milik auth-server yang menangani mekanisme otorisasi yaitu "http://localhost:8100/acl", dengan data topic yang bernilai "nodeMCU/DHT/temperature/status", acc yang bernilai "2" yang manyatakan bahwa ini merupakan pesan publish serta clientid milik klien. Setelah itu diterima respon dari auth-server terkait pesan tersebut, dan dinyatakan bahwa pengguna tersebut tidak terotorisasi untuk melakukan publish di topik tersebut, yang dinyatakan dengan "AUTHORIZED=0". Lalu broker melakukan "Denied PUBLISH" pesan tersebut. Konfigurasi dengan auth-server ini berhasil menolak pesan publish yang dikirimkan pada topik yang terdapat pada topic tree namun diluar dari hak akses milik pengguna yang diperiksa pada ACL di database. Dikarenakan pengguna memiliki peran tertentu, dan pada peran tersebut telah dirancang tiap topik yang dapat diakses, maka broker berhasil menolak publish diluar hak akses milik pengguna tersebut. Hasil yang didapatkan oleh broker pada skenario ini dama dengan yang dilakukan pada skenario PMOP 202, karena mekanisme autentikasi yang dibuat pada auth-server tidak diubah.



Gambar 6.29 Screenshot Broker PMOP_302

Seperti yang terlihat pada gambar 6.30, bahwa hasil *sniffing interface* didapatkan tidak ditemukan protokol MQTT, hal ini karena pertukaran paket data dilakukan melalui TLS. Hasil *sniffing* tidak mendapatkan pesan *PUBLISH* yang dikirimkan oleh klien ke broker, namun jika dilihat berdasarkan paket yang dikirim dari ip klien menuju ip broker dapat dilihat pertukaran paket *Application Data*. Jika dilihat pada isi data dari paket *Application Data* tersebut dapat dilihat bahwa data yang dikirimkan dapat tidak dapat dibaca karena dalam bentuk enkripsi data.

0	9 🗧		*any																
E	ile	<u>E</u> dit	<u>V</u> iew	<u>G</u> o	<u>C</u> aptu	re <u>A</u> r	alyze	<u>S</u> tati	istics	Telep	hon <u>y</u>	Wire	eless	<u>T</u> oo	ls <u>H</u> e	elp			
			6	۲	01 01 01		6	9			Ì	1	₽			Ð	Q	9	•
	Ap	oply a	display	y filter	r <ct< th=""><th>rl-/></th><th></th><th></th><th></th><th></th><th></th><th></th><th></th><th></th><th></th><th></th><th>•</th><th>Exp</th><th>oressio</th></ct<>	rl-/>											•	Exp	oressio
No).	Time	Sour	ce		D	estinat	ion		Prot	ocol	Lengt	h Info						
	1	0.0	192.	168.1	L00.24	22	4.0.0	.251		MDN	S	105	5 Sta	ndarc	i quer	y 0x0	003 P	TR _	80574:
	2	0.2	192.	168.1	L00.11	22	4.0.0	.251		MDN	S	105	5 Sta	ndard	i quer	ý 0x0	003 P	TR _	233637
L C	3	3.8	192.	168.1	100.29	19	2.168	.100.	40	TLS	/1.2	149) App	licat	ion D	ata			
	4	3.8	192.	168.1	LOO.40	19	2.168	.100.	29	TCP		56	5 888	3 → 6	61964	[ACK]	Seq=	1 Ac	k=94 N
	5	3.8	127.	0.0.1	L	12	27.0.0	.1		TCP		76	5 393	80 →	8100	[SYN]	Seq=	0 Wi	n=4369
►	Fr	ame 3	: 149	byte	s on wi	ire (1	192 b:	its),	149 H	oytes	capti	ured ((1192	bit	s) on	inter	face	0	
►	Li	nux co	ooked	capt	ure														
▶	In	ternet	t Prot	tocol	Versi	on 4,	Src: :	192.10	58.100	9.29,	Dst:	192.1	168.1	.00.4	9				
►	Tr	ansmi	ssion	Cont	rol Pro	otocol	, Src	Port	: 6190	64, Ds	t Poi	rt: 88	383,	Seq:	1, Ac	:k: 1,	Len:	93	
•	Se	cure S	Socket	ts La	yer				_										
	•	TLSv1	L.2 Re	cord	Layer:	: Appl	LCatio	on Dat	a Pro	tocol	: mqt	t							
•		Co	ntent	Type	: Abbī	lcat1	on Dat	a (23)										
		ve	rsion	: 115	5 1.2 (0X030	5)												
		Le	ngch:	88 ad A r	nlicat	ion D	+	00000	00000	00000	22206	7. f	£0.20	cho0o	dd745	0fbd4	2		
		En	crypt	ea Ap	pricat	ION Da	ita: 6	00000	00000	00002	5320D	refca	T92C	obaea	laa/4T	81004	zea	•	

Gambar 6.30 Hasil Capture Wireshark PMOP_302

6.2.2.8 Klien Melakukan *Publish* Pada Topik Yang Sesuai Dengan Hak Akses Di *Topic Tree* Yang Dibuat. (Kode: PMOP_303)

Pada skenario pengujian ini digunakan konfigurasi broker MQTT yang akan menerapkan Auth-Server menggunakan *backend* JWT dan menerapkan keamanan TLS. Prosedur pengujian pada skenario ini akan dimulai dengan melakukan *capture interfaces* di Wireshark lalu dengan aplikasi MQTT-Spy klien akan mengirim pesan *PUBLISH* pada topik yang terdapat dalam *topic tree* dan sesuai dengan hak akses milik pengguna tersebut ke broker, setelah itu akan dilihat status broker, selanjutnya akan dilakukan *filter* pada hasil *capture interface* menggunakan *field* MQTT yang akan menyeleksi setiap paket yang dikirimkan menggunakan protokol MQTT untuk melihat hasil *sniffing*. Pada pengujian ini digunakan pengguna yang terdaftar dalam sistem yaitu siMerah yang memiliki peran sebagai red_user.

Pada gambar 6.31, dapat dilihat bahwa broker menerima sebuah pesan *PUBLISH* kemudian mengirimkan pesan tersebut ke url milik auth-server yang menangani mekanisme otorisasi yaitu "http://localhost:8100/acl", dengan data *topic* yang bernilai "nodeMCU/LED/red/toggle", acc yang bernilai "2" yang manyatakan bahwa ini merupakan pesan *publish* serta clientid milik klien. Setelah itu diterima respon dari auth-server terkait pesan tersebut, dan dinyatakan bahwa pengguna tersebut terotorisasi untuk melakukan *publish* di topik tersebut, yang dinyatakan dengan "*AUTHORIZED*=1". Lalu broker melakukan "*Received PUBLISH*" pesan tersebut, yang menandakan bahwa pesan publish berhasil diterima dan akan diteruskan pada *subscriber* pada topik tersebut. Konfigurasi dengan auth-server ini berhasil menerima pesan *publish* yang dikirimkan pada topik yang terdapat pada *topic tree* dan sesuai dengan hak akses milik pengguna yang diperiksa pada ACL di database. Hasil yang didapatkan oleh broker pada skenario ini dama dengan yang dilakukan pada skenario PMOP_203, karena mekanisme autentikasi yang dibuat pada auth-server tidak diubah.

1509784244: url=http://localhost:8100/acl
1509784244: data=topic=nodeMCU%2FLED%2Fred%2Ftoggle&
acc=2&clientid=Klien_Pengujian
1509784244: aclcheck(eyJhbGciOiJIUzI1NiIsInR5cCI6Ikp
XVCJ9.eyJ1c2VybmFtZSI6InNpTWVyYWgiLCJwdyI6InJlZDEyMzQ1In
0.vvcTUByumkomj-oxQpqrGVkjK89zncElr4kFKxgkIQI, nodeMCU/L
ED/red/toggle, 2) trying to acl with jwt
1509784244: aclcheck(eyJhbGciOiJIUzI1NiIsInR5cCI6Ikp
XVCJ9.eyJ1c2VybmFtZSI6InNpTWVyYWgiLCJwdyI6InJlZDEyMzQ1In
0.vvcTUByumkomj-oxQpqrGVkjK89zncElr4kFKxgkIQI, nodeMCU/L
ED/red/toggle, 2) AUTHORIZED=1 by jwt
1509784244: Cached [5B8F665F3B22379223AEA0EBDFBA01
83CADE5F86] for (Klien_Pengujian,eyJhbGciOiJIUzI1NiIsInR
5cCI6IkpXVCJ9.eyJ1c2VybmFtZSI6InNpTWVyYWgiLCJwdyI6InJlZD
EyMzQ1In0.vvcTUByumkomj-oxQpqrGVkjK89zncElr4kFKxgkIQI,2)
1509784244: Received PUBLISH from Klien_Pengujian (d0, q
0, r0, m0, 'nodeMCU/LED/red/toggle', (30 bytes))

Gambar 6.31 Screenshot Broker PMOP_303

Seperti yang terlihat pada gambar 6.30, bahwa hasil *sniffing interface* didapatkan tidak ditemukan protokol MQTT, hal ini karena pertukaran paket data dilakukan melalui TLS. Hasil *sniffing* tidak mendapatkan pesan *PUBLISH* yang dikirimkan oleh klien ke broker, namun jika dilihat berdasarkan paket yang dikirim dari ip klien menuju ip broker dapat dilihat pertukaran paket *Application Data*. Jika dilihat pada isi data dari paket *Application Data* tersebut dapat dilihat bahwa data yang dikirimkan dapat tidak dapat dibaca karena dalam bentuk enkripsi data.

8	•	•	*any											
<u>F</u> il	e	Edit	<u>V</u> iew	<u>G</u> o	<u>C</u> apture	<u>A</u> nalyze	<u>S</u> tatistics	Telephon <u>y</u>	<u>W</u> irele	ss <u>T</u> oo	ls <u>H</u> elp)		
			6	9	0107 0110 0111	8	۹ 🔶	• 🔿 🖀	<u></u>	Ł 🜉		Ð (e , e	
	Ap	oly a	display	/ filter	<ctrl-></ctrl->	>							- E>	cpressic
No.		Time	Sourc	ce		Destinat	ion	Protocol	Length Ir	nfo				
	1	0.0	192.	168.1	.00.24	224.0.0	.251	MDNS	105 S	Standard	d query	0x000	3 PTR	80574
	2	0.2	192.	168.1	.00.11	224.0.0	.251	MDNS	105 S	Standard	i query	0x000	3 PTR	_23363
Г	3	0.3	192.	168.1	.00.29	192.168	.100.40	TLSv1.2	141 A	<pre>\pplicat</pre>	ion Dat	ta		
L	4	0.3	192.	168.1	00.40	192.168	.100.29	TCP	56 8	3883 → 6	61965 [A	ACK] S	eq=1 A	ck=86
	5	15	Sams	unge_	/1:/4:12			ARP	02 W	no nas	192.108	3.100.	17 Tel	1 192.
	Fra	me 3	: 141	bytes	s on wire	(1128 b:	its), 141	bytes capt	ured (11	128 bit:	s) on i	nterfa	ace 0	
	Lin	ux co	poked	capti	ire									
2	Int	erne	t Prot	Cont.	version 4	4, Src: :	192.168.10	0.29, Dst:	192.168	8.100.40	9 1 Aole			
-	Sec	IISIIII:	Sockot	CONLI	OI Proto	501, SFC	Port: 019	05, DSL PO	rt: 000	s, seq:	I, ACK	: 1, l	en: ac)
	-	TLSv1	2 Re	cord	laver: Ar	nlicatio	n Data Pr	otocol: mat	++					
	•	Co	ntent	Type	: Applica	tion Dat	a (23)	ococor. mq						
		Ve	rsion	: TLS	1.2 (0x0	303)								
		Le	ngth:	80		,								
		En	crypt	ed Ap	plication	Data: 0	0000000000	00000376888	32b76b7b	66c770a	l6da0e3b	of133c	9	
			21	•	-									

Gambar 6.32 Hasil Capture Wireshark PMOP_303

6.2.3 Pengujian Mekanisme Otorisasi Sistem Melalui Subscribe

Pengujian ini dilakukan untuk mengetahui hasil penerapan mekanisme otorisasi pada sistem berbasis protokol MQTT pada saat klien melakukan *subscribe* untuk menerima pesan *publish* ke broker. Pengujian ini akan membandingkan bagaimana perbedaan keamanan pada ketiga konfigurasi broker MQTT yang sudah dijabarkan, ketiga konfigurasi broker akan memiliki beberapa skenario yang akan dilakukan. Pengujian akan dilakukan dengan cara klien MQTT yang sudah terkoneksi dengan broker dan berhasil terautentikasi, akan mengirimkan pesan *Publish* ke broker MQTT, kemudian pada tiap skenario akan digunakan parameter yang berbeda terkait topik yang akan digunakan, apakah dengan sesuai dengan *topic tree*, atau tidak dan sesuai dengan hak akses atau tidak.

Karena pemeriksaan klien MQTT *subscribe* pada broker mosquitto akan dilakukan setiap ada pesan yang di-*publish* pada topik tersebut, maka pengujian ini kan dilakukan dengan menggunakan suatu klien yang akan melakukan *publish* sesuai dengan topik yang di-*subscribe* oleh klien lainnya. Pada pengujian ini, pertama-tama Wireshark akan dijalankan dan melakukan *capture* kemudian klien MQTT akan mengirimkan pesan *subscribe* ke broker MQTT dengan topik yang disesuaikan pada tiap skenario, lalu klien MQTT lainnya akan melakukan *publish* sesuai dengan topik klien MQTT yang *subscribe*. Pesan *subscribe* dan *publish* akan dikirimkan menggunakan aplikasi dashboard MQTT-spy dengan menggunakan data klien yang terdaftar dalam database, setelah dilakukan pengiriman pesan *publish* dari klien MQTT kedua selanjutnya akan dilihat apakah klien MQTT yang status dari mosquitto-broker dan juga auth-server jika digunakan.

6.2.3.1 Klien Melakukan *Subscribe* Pada Topik Yang Tidak Ada Dalam *Topic Tree* Yang Dibuat. (Kode: PMOS_101)

Pada skenario pengujian ini digunakan konfigurasi broker MQTT yang tidak menerapkan Auth-Server dan tidak menerapkan TLS. Prosedur pengujian pada skenario ini akan dimulai dengan melakukan *capture interfaces* di Wireshark, lalu dengan aplikasi MQTT-Spy klien pertama akan mengirim pesan *SUBSCRIBE* pada topik yang tidak terdapat dalam *topic tree* ke broker, kemudian klien kedua akan mengirimkan pesan *PUBLISH* yang disesuaikan dengan klien pertama, setelah itu akan dilihat status broker, selanjutnya akan dilakukan *filter* pada hasil *capture interface* menggunakan *field* MQTT yang akan menyeleksi setiap paket yang dikirimkan menggunakan protokol MQTT untuk melihat hasil *sniffing*.

Pada gambar 6.33, dapat dilihat bahwa broker meneruskan pesan PUBLISH dari klien kedua ke klien pertama, dengan menggunakan konfigurasi ini klien pertama dapat *subscribe* pada topik manapun baik yang dirancang pada *topic tree* maupun diluar dari rancangan, sehingga dibutuhkan sebuah mekanisme otorisasi untuk mengatur hak akses pengguna dan topik yang dapat diakses.

1509787890: Sending PUBLISH to Klien_Subscribe (d0, q0, r0, m0, 'topik/pengujian/apaSaja', ... (31 bytes))

Gambar 6.33 Screenshot Broker PMOS_101

Seperti yang terlihat pada gambar 6.34, bahwa hasil *sniffing* protokol MQTT mendapatkan dua pesan *PUBLISH* yang pertama merupakan pesan yang dikirimkan oleh klien kedua ke broker, lalu yang kedua merupakan yang diteruskan oleh broker menuju ke klien pertama. Jika dilihat pada isi data dari kedua paket *publish* tersebut adalah sama dan dapat dilihat bahwa data yang dikirimkan dapat dibaca dalam bentuk *plain text*, kemudian jika dilihat isi dari pesan *publish* tersebut didalamnya terdapat *field Topic* yang memiliki nilai

"topik/pengujian/apaSaja" dengan *filed Message* yang memiliki nilai "*publish* topik diluar *topic tree*".



Gambar 6.34 Hasil Capture Wireshark PMOS_101

6.2.3.2 Klien Melakukan *Subscribe* Pada Topik Yang Ada Dalam *Topic Tree* Yang Dibuat. (Kode: PMOS_102)

Pada skenario pengujian ini digunakan konfigurasi broker MQTT yang tidak menerapkan Auth-Server dan tidak menerapkan TLS. Prosedur pengujian pada skenario ini akan dimulai dengan melakukan *capture interfaces* di Wireshark, lalu dengan aplikasi MQTT-Spy klien pertama akan mengirim pesan *SUBSCRIBE* pada topik yang terdapat dalam *topic tree* ke broker, kemudian klien kedua akan mengirimkan pesan *PUBLISH* yang disesuaikan dengan klien pertama, setelah itu akan dilihat status broker, selanjutnya akan dilakukan *filter* pada hasil *capture interface* menggunakan *field* MQTT yang akan menyeleksi setiap paket yang dikirimkan menggunakan protokol MQTT untuk melihat hasil *sniffing*.

Pada gambar 6.35, dapat dilihat bahwa broker meneruskan pesan PUBLISH dari klien kedua ke klien pertama, dengan menggunakan konfigurasi ini klien pertama dapat *subscribe* pada topik manapun baik yang dirancang pada *topic tree* maupun diluar dari rancangan, sehingga dibutuhkan sebuah mekanisme otorisasi untuk mengatur hak akses pengguna dan topik yang dapat diakses.

1509788208: Sending PUBLISH to Klien_Subscribe (d0, q0, r0, m0, 'nodeMCU/DHT/temperature/status', ... (29 bytes)

Gambar 6.35 Screenshot Broker PMOS_102

Seperti yang terlihat pada gambar 6.36, bahwa hasil *sniffing* protokol MQTT mendapatkan dua pesan *PUBLISH* yang pertama merupakan pesan yang dikirimkan oleh klien kedua ke broker, lalu yang kedua merupakan yang diteruskan oleh broker menuju ke klien pertama. Jika dilihat pada isi data dari kedua paket *publish* tersebut adalah sama dan dapat dilihat bahwa data yang dikirimkan dapat dibaca dalam bentuk *plain text*, kemudian jika dilihat isi dari pesan *publish*

tersebut didalamnya terdapat *field Topic* yang memiliki nilai "nodeMCU/DHT/temperature/status" dengan *field Message* yang memiliki nilai "*publish* topik pada *topic tree*". Dari hasil yang didapatkan pada skenario ini adalah sama dengan yang didapatkan pada skenario PMOS_101.

	3 (•	any																	
F	ile	<u>E</u> dit	<u>V</u> iew	<u>G</u> o	<u>C</u> apture	<u>A</u> naly	/ze	Stati	stics	Telep	bhony	<u>W</u> ir	eless	Too	ols <u>I</u>	<u>H</u> elp				
4			1		0100 0100 01100 01110	×	6	9			Ď	1	垫				Ð	Q	€	
	m	qtt															×]-	•	Exp	ressio
No	o.	Time	Source	e		Dest	inatio	on		Pro	tocol	Leng	th Info)						
	1	0.0	192.1	68.1	00.29	192.	168.	100.4	40	MQT	Т	11	9 Pub	lish	Mes	sage				
	2	0.0	192.1	.68.1	00.40	192.	168.	100.2	29	MÕT	Т	11	9 Pub	lish	Mes	sage				
	7	2.5	192.1	.68.1	00.29	192.	168.	100.4	40	MQT	Т	6	2 Pir	ng Re	ques	t				
	g	2.5	192.1	.68.1	00.40	192.	168.	100.2	29	MQT	Т	5	8 Pir	ng Re	spon	se				
* * * *	Fr Li In Tr MQ	ame 2: nux cc ternet ansmis Telen Publi	119 b oked c Proto sion (netry 1 sh Mes	oytes captu ocol Contr Frans sage	s on wire Tre Version Tol Prote Sport Pre	e (952 4, Sro ocol, S otocol	bit: c: 19 Src F	s), 1 92.16 Port:	19 by 8.100 1883	/tes (0.40, 3, Dst	Dst: Dst: Por	red (192. t: 62	952 168.: 074,	bits) 100.2 Seq:	on 9 1,	inte Ack:	erfa 1,	ce 0 Len:	63	
	 Publish Message 0011 0000 = Header Flags: 0x30 (Publish Message) Msg Len: 61 Topic: nodeMCU/DHT/temperature/status Message: publish topik pada topic tree 																			

Gambar 6.36 Hasil Capture Wireshark PMOS_102

6.2.3.3 Klien Melakukan *Subscribe* Pada Topik Diluar *Topic Tree* Yang Dibuat. (Kode: PMOS 201)

Pada skenario pengujian ini digunakan konfigurasi broker MQTT yang tidak menerapkan Auth-Server dan tidak menerapkan TLS. Prosedur pengujian pada skenario ini akan dimulai dengan melakukan *capture interfaces* di Wireshark, lalu dengan aplikasi MQTT-Spy klien pertama akan mengirim pesan *SUBSCRIBE* pada topik yang tidak terdapat dalam *topic tree* ke broker, kemudian klien kedua akan mengirimkan pesan *PUBLISH* yang disesuaikan dengan klien pertama, setelah itu akan dilihat status broker, selanjutnya akan dilakukan *filter* pada hasil *capture interface* menggunakan *field* MQTT yang akan menyeleksi setiap paket yang dikirimkan menggunakan protokol MQTT untuk melihat hasil *sniffing*. Pada pengujian ini digunakan pengguna yang terdaftar dalam sistem yaitu siSuhu yang memiliki peran sebagai temperature_user.

Pada gambar 6.37, dapat dilihat bahwa broker sebelum meneruskan pesan yang di-*publish* oleh klien kedua akan mengirimkan sebuah *request* ke url milik auth-server yang menangani mekanisme otorisasi yaitu pada alamat "http://localhost:8100/acl", dengan memeberikan data *topic* yang bernilai "topik/pengujian/apaSaja", acc yang bernilai "1" yang manyatakan bahwa ini merupakan pesan *subscribe* serta clientid milik klien pertama. Setelah itu diterima respon dari auth-server terkait pesan tersebut, dan dinyatakan bahwa pengguna tersebut tidak terotorisasi untuk menerima / *subscribe* pesan yang di-*publish* di topik tersebut, yang dinyatakan dengan "*AUTHORIZED*=0". Lalu broker tidak akan meneruskan *publish* tersebut ke klien pertama. Konfigurasi dengan auth-server ini berhasil menolak pesan *subscribe* yang dikirimkan pada topik yang tidak terdapat pada *topic tree* yang diperiksa pada ACL di database. Dikarenakan setiap topik

yang dapat diakses sudah dirancang sebelumnya sehingga pengguna tidak dapat mengakses topik diluar dari perancancangan topik tersebut.



Gambar 6.37 Screenshot Broker PMOS_201

Seperti yang terlihat pada gambar 6.38, bahwa hasil *sniffing* protokol MQTT mendapatkan pesan satu *PUBLISH* yang dikirimkan oleh klien kedua ke broker, namun tidak didapatkan pesan *PUBLISH* yang diteruskan oleh broker ke klien pertama. Hal ini menandakan bahwa tidak ada pesan yang diteruskan oleh broker ke klien pertama karena ia tidak memiliki hak akses untuk mendapatkan pesan tersebut.

		•	any																
E	le	<u>E</u> dit	<u>V</u> iew	<u>G</u> o	<u>C</u> apture	<u>A</u> nalyz	e <u>S</u> tat	istics	Telep	hony	<u>W</u> ir	eless	<u>T</u> oo	ls <u>H</u>	lelp				
(6		0101 0110 0110	8				Ì	•	₹			€		2	€	
	ma	qtt													X	(→	•	Expr	essio
No		Time	Sourc	e		Destin	ation		Prot	ocol	Leng	th Info							
E.	3	4.8	192.1	168.1	.00.29	192.1	68.100.	40	MQT	Т	10	8 Pub	lish	Mess	age				
	70	33	192.1	168.1	.00.29	192.1	68.100.	40	MQT	Т	8	5 Uns	ubsci	ribe	Requ	est			
	71	33	192.3	168.1	.00.40	192.1	68.100.	29	MÕT	Т	6	0 Uns	ubscr	ribe	Ack				
	72	33	192.1	168.1	.00.29	192.1	68.100.	40	MÕT	Т	6	2 Dis	conne	ect R	lea				
	77	34	192.3	168.1	.00.29	192.1	68.100.	40	MQT	Т	6	2 Dis	conne	ect R	teq				
•	Fra	ame 3:	108	bytes	s on wire	(864 k	its), :	108 by	/tes c	aptu	red (864 k	oits)	oni	inter	face	0		
•	Lin	iux co	oked	capti	ire				·										
* * *	Int Tra MQ	ernet Insmis Telen	: Prot ssion netry	ocol Conti Trans	Version rol Proto sport Pro	4, Src: col, Sr tocol	192.1 c Port	68.100 : 6209	9.29, 97, Ds	Dst: st Po	192. rt: 1	168.1 1883,	LOO.4 Seq:	0 1, /	Ack:	1, L	en:	52	

Gambar 6.38 Hasil Capture Wireshark PMOS_201

6.2.3.4 Klien Melakukan *Subscribe* Pada Topik Yang Tidak Sesuai Dengan Hak Akses Di *Topic Tree* Yang Dibuat. (Kode: PMOS_202)

Pada skenario pengujian ini digunakan konfigurasi broker MQTT yang tidak menerapkan Auth-Server dan tidak menerapkan TLS. Prosedur pengujian pada skenario ini akan dimulai dengan melakukan *capture interfaces* di Wireshark, lalu dengan aplikasi MQTT-Spy klien pertama akan mengirim pesan *SUBSCRIBE* pada topik yang terdapat dalam *topic tree* namun diluar dari hak akses milik pengguna tersebut ke broker, kemudian klien kedua akan mengirimkan pesan *PUBLISH* yang disesuaikan dengan klien pertama, setelah itu akan dilihat status broker, selanjutnya akan dilakukan *filter* pada hasil *capture interface* menggunakan *field* MQTT yang akan menyeleksi setiap paket yang dikirimkan menggunakan protokol MQTT untuk melihat hasil *sniffing*. Pada pengujian ini digunakan pengguna yang terdaftar dalam sistem yaitu siSuhu yang memiliki peran sebagai temperature_user. Pada gambar 6.39, dapat dilihat bahwa broker sebelum meneruskan pesan yang di-*publish* oleh klien kedua akan mengirimkan sebuah *request* ke url milik auth-server yang menangani mekanisme otorisasi yaitu pada alamat "http://localhost:8100/acl", dengan memberikan data *topic* yang bernilai "nodeMCU/LED/red/status", acc yang bernilai "1" yang manyatakan bahwa ini merupakan pesan *subscribe* serta clientid milik klien pertama. Setelah itu diterima respon dari auth-server terkait pesan tersebut, dan dinyatakan bahwa pengguna tersebut tidak terotorisasi untuk menerima / *subscribe* pesan yang di-*publish* di topik tersebut, yang dinyatakan dengan "*AUTHORIZED=0*". Lalu broker tidak akan meneruskan *publish* tersebut ke klien pertama. Konfigurasi dengan auth-server ini berhasil menolak pesan *subscribe* yang dikirimkan pada topik yang terdapat pada *topic tree* namun diluar dari hak akses milik pengguna yang diperiksa pada ACL di database. Dikarenakan pengguna memiliki peran tertentu, dan pada peran tersebut telah dirancang tiap topik yang dapat diakses, maka broker berhasil menolak *publish* diluar hak akses milik pengguna tersebut.



Gambar 6.39 Screenshot Broker PMOS_202

Seperti yang terlihat pada gambar 6.40, bahwa hasil *sniffing* protokol MQTT mendapatkan pesan satu *PUBLISH* yang dikirimkan oleh klien kedua ke broker, namun tidak didapatkan pesan *PUBLISH* yang diteruskan oleh broker ke klien pertama. Hal ini menandakan bahwa tidak ada pesan yang diteruskan oleh broker ke klien pertama karena ia tidak memiliki hak akses untuk mendapatkan pesan tersebut.

8		0	Captu	uring	from	any															
<u> </u>	ile	<u>E</u> dit	<u>V</u> iew	<u>G</u> o	<u>C</u> ap	ture	<u>A</u> na	lyze	<u>S</u> tati	stics	Tele	phony	<u>W</u> i	reless	Too	ls <u>F</u>	<u>l</u> elp				
[1		8	۲		0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.0	X	6	9	¢	•		1	₹			(Ð	Q	⊜,) (
	m	qtt																×]-	•	Exp	ressio
No		Time	Sour	rce			Des	tinati	ion		Pro	tocol	Leng	th Info	5						
Г	3	2.1.	. 192	.168.	100.2	9	192	.168	.100.	40	MQ1	T	10)6 Pul	olish	Mess	sage				
•	Fr	ame 3	: 106	byte	es on	wire	(848	8 bit	:s), 1	106 b	ytes	captu	red ((848	bits)	on	inte	erfac	ce O		
•	Li	nux c	ooked	capt	ure																
	In Tr MQ	terne ansmi Tele	t Pro ssion metry	tocol Cont Tran	l Vers rol P sport	ion roto Pro	4, Sı col, tocol	rc: 1 Src l	.92.10 Port:	68.10 621	9.29, 99, D	Dst: st Po	192. rt: 1	.168. L883,	100.4 Seq:	0 1,	Ack:	1,	Len:	50	

Gambar 6.40 Hasil Capture Wireshark PMOS_202

6.2.3.5 Klien Melakukan *Subscribe* Pada Topik Yang Sesuai Dengan Hak Akses Di *Topic Tree* Yang Dibuat. (Kode: PMOS_203)

Pada skenario pengujian ini digunakan konfigurasi broker MQTT yang tidak menerapkan Auth-Server dan tidak menerapkan TLS. Prosedur pengujian pada skenario ini akan dimulai dengan melakukan *capture interfaces* di Wireshark, lalu dengan aplikasi MQTT-Spy klien pertama akan mengirim pesan *SUBSCRIBE* pada topik yang terdapat dalam *topic tree* dan sesuai dengan hak akses milik pengguna tersebut ke broker, kemudian klien kedua akan mengirimkan pesan *PUBLISH* yang disesuaikan dengan klien pertama, setelah itu akan dilihat status broker, selanjutnya akan dilakukan *filter* pada hasil *capture interface* menggunakan *field* MQTT yang akan menyeleksi setiap paket yang dikirimkan menggunakan protokol MQTT untuk melihat hasil *sniffing*. Pada pengujian ini digunakan pengguna yang terdaftar dalam sistem yaitu siSuhu yang memiliki peran sebagai temperature_user.

Pada gambar 6.41, dapat dilihat bahwa broker sebelum meneruskan pesan yang di-*publish* oleh klien kedua akan mengirimkan sebuah *request* ke url milik auth-server yang menangani mekanisme otorisasi yaitu pada alamat "http://localhost:8100/acl", dengan memberikan data *topic* yang bernilai "nodeMCU/DHT/temperature/status", acc yang bernilai "1" yang manyatakan bahwa ini merupakan pesan *subscribe* serta clientid milik klien pertama. Setelah itu diterima respon dari auth-server terkait pesan tersebut, dan dinyatakan bahwa pengguna tersebut terotorisasi untuk menerima / *subscribe* pesan yang di-*publish* di topik tersebut, yang dinyatakan dengan "*AUTHORIZED*=1". Lalu broker akan meneruskan *publish* tersebut ke klien pertama. Konfigurasi dengan auth-server ini berhasil meneruskan pesan *publish* ke klien pertama yang *subscribe* pada topik yang terdapat pada *topic tree* dan sesuai dengan hak akses milik pengguna tersebut yang diperiksa pada ACL di database.



Gambar 6.41 Screenshot Broker PMOS_203

Seperti yang terlihat pada gambar 6.40, bahwa hasil *sniffing* protokol MQTT mendapatkan dua pesan *PUBLISH* yang dikirimkan oleh klien kedua ke broker dan yang diteruskan oleh broker ke klien pertama. Hal ini menandakan bahwa klien pertama berhasil *subscribe* karena ia memiliki hak akses untuk mendapatkan pesan tersebut. Jika dilihat pada isi data dari paket *publish* tersebut dapat dilihat bahwa data yang dikirimkan dapat dibaca dalam bentuk *plain text*, kemudian jika dilihat isi dari pesan *publish* tersebut didalamnya terdapat *field Topic* yang

memiliki nilai "nodeMCU/DHT/temperature/status" dengan *filed Message* yang memiliki nilai "*publish* sesuai ACL". Meskipun berdasarkan keberhasilan mekanisme otorisasi pada pengujian PMOS_201, PMOS_202 dan PMOS_203, namun hasil *sniffing* berhasil mendapatkan topik dan pesan yang dipublish oleh klien pada sistem.



Gambar 6.42 Hasil Capture Wireshark PMOS_203

6.2.3.6 Klien Melakukan *Subscribe* Pada Topik Diluar *Topic Tree* Yang Dibuat. (Kode: PMOS_301)

Pada skenario pengujian ini digunakan konfigurasi broker MQTT yang tidak menerapkan Auth-Server dan menerapkan keamanan TLS. Prosedur pengujian pada skenario ini akan dimulai dengan melakukan *capture interfaces* di Wireshark, lalu dengan aplikasi MQTT-Spy klien pertama akan mengirim pesan *SUBSCRIBE* pada topik yang tidak terdapat dalam *topic tree* ke broker, kemudian klien kedua akan mengirimkan pesan *PUBLISH* yang disesuaikan dengan klien pertama, setelah itu akan dilihat status broker, selanjutnya akan dilakukan *filter* pada hasil *capture interface* menggunakan *field* MQTT yang akan menyeleksi setiap paket yang dikirimkan menggunakan protokol MQTT untuk melihat hasil *sniffing*. Pada pengujian ini digunakan pengguna yang terdaftar dalam sistem yaitu siSuhu yang memiliki peran sebagai temperature user.

Pada gambar 6.43, dapat dilihat bahwa broker sebelum meneruskan pesan yang di-*publish* oleh klien kedua akan mengirimkan sebuah *request* ke url milik auth-server yang menangani mekanisme otorisasi yaitu pada alamat "http://localhost:8100/acl", dengan memeberikan data *topic* yang bernilai "topik/pengujian/apaSaja", acc yang bernilai "1" yang manyatakan bahwa ini merupakan pesan *subscribe* serta clientid milik klien pertama. Setelah itu diterima respon dari auth-server terkait pesan tersebut, dan dinyatakan bahwa pengguna tersebut tidak terotorisasi untuk menerima / *subscribe* pesan yang di-*publish* di topik tersebut, yang dinyatakan dengan "AUTHORIZED=0". Lalu broker tidak akan meneruskan *publish* tersebut ke klien pertama. Konfigurasi dengan auth-server ini

berhasil menolak pesan *subscribe* yang dikirimkan pada topik yang tidak terdapat pada *topic tree* yang diperiksa pada ACL di database. Dikarenakan setiap topik yang dapat diakses sudah dirancang sebelumnya sehingga pengguna tidak dapat mengakses topik diluar dari perancancangan topik tersebut. Hasil yang didapatkan oleh broker pada skenario ini dama dengan yang dilakukan pada skenario PMOS_201, karena mekanisme autentikasi yang dibuat pada auth-server tidak diubah.

1509789968: url=http://localhost:8100/acl
1509789968: data=topic=topik%2Fpengujian%2FapaSaja&a
cc=1&clientid=Klien Subscribe
1509789968: aclcheck(eyJhbGciOiJIUzI1NiIsInR5cCI6Ikp
XVCJ9.eyJ1c2VybmFtZSI6InNpU3VodSIsInB3IjoidGVtcDEyMzOifO
.P4iOiv5hJIbtfIF01-nrHqTJJqOPeuBf44NxO wu Dq, topik/penq
ujian/apaSaja, 1) AUTHORIZED=0 by (null)

Gambar 6.43 Screenshot Broker PMOS_301

Seperti yang terlihat pada gambar 6.44, bahwa hasil *sniffing interface* didapatkan tidak ditemukan protokol MQTT, hal ini karena pertukaran paket data dilakukan melalui TLS. Hasil *sniffing* tidak mendapatkan pesan *PUBLISH* yang dikirimkan oleh klien ke broker, namun jika dilihat berdasarkan paket yang dikirim dari ip klien menuju ip broker dapat dilihat pertukaran satu paket *Application Data* ini merupakan pesan *PUBLISH* yang dikirimkan oleh klien ke broker, namun tidak didapatkan pesan *PUBLISH* yang diteruskan oleh klien kedua ke broker, namun tidak didapatkan pesan *PUBLISH* yang diteruskan oleh broker ke klien pertama. Hal ini menandakan bahwa tidak ada pesan yang diteruskan oleh broker ke klien pertama karena ia tidak memiliki hak akses untuk mendapatkan pesan tersebut. Jika dilihat pada isi data dari paket *Application Data* tersebut dapat dilihat bahwa data yang dikirimkan dapat tidak dapat dibaca karena dalam bentuk enkripsi data.

8		*any	/																
<u> </u>	le <u>E</u> di	t <u>V</u> ie	w <u>G</u> o	<u>C</u> aptu	ire <u>A</u>	<u>A</u> nalyze	<u>S</u> tati	stics	Telep	hony	<u>W</u> ir	eless	Too	ols <u>H</u>	lelp				
		5	۲			3	9				1	垫			9	Ð	Q	٩	1
	ssl														Þ	<] -	•	Exp	ression
No	. Tin	ne So	urce		[Destinat	ion		Prot	ocol	Lengt	h Info)						
Г	2 3.5	5 19	2.168.	100.29	1	192.168	3.100.	40	TLS	v1.2	13	7 App	olica	tion	Data	a 🛛			

* * * * *	 Frame 2: 137 bytes on wire (1096 bits), 137 bytes captured (1096 bits) on interface 0 Linux cooked capture Internet Protocol Version 4, Src: 192.168.100.29, Dst: 192.168.100.40 Transmission Control Protocol, Src Port: 62138, Dst Port: 8883, Seq: 1, Ack: 1, Len: 81 Secure Sockets Layer 																		

Gambar 6.44 Hasil Capture Wireshark PMOS_301

6.2.3.7 Klien Melakukan *Subscribe* Pada Topik Yang Tidak Sesuai Dengan Hak Akses Di *Topic Tree* Yang Dibuat. (Kode: PMOS_302)

Pada skenario pengujian ini digunakan konfigurasi broker MQTT yang tidak menerapkan Auth-Server dan menerapkan keamanan TLS. Prosedur pengujian pada skenario ini akan dimulai dengan melakukan *capture interfaces* di Wireshark, lalu dengan aplikasi MQTT-Spy klien pertama akan mengirim pesan *SUBSCRIBE* pada topik yang terdapat dalam *topic tree* namun diluar dari hak akses milik pengguna tersebut ke broker, kemudian klien kedua akan mengirimkan pesan *PUBLISH* yang disesuaikan dengan klien pertama, setelah itu akan dilihat status broker, selanjutnya akan dilakukan *filter* pada hasil *capture interface* menggunakan *field* MQTT yang akan menyeleksi setiap paket yang dikirimkan menggunakan protokol MQTT untuk melihat hasil *sniffing*. Pada pengujian ini digunakan pengguna yang terdaftar dalam sistem yaitu siSuhu yang memiliki peran sebagai temperature_user.

Pada gambar 6.45, dapat dilihat bahwa broker sebelum meneruskan pesan yang di-publish oleh klien kedua akan mengirimkan sebuah request ke url milik auth-server yang menangani mekanisme otorisasi yaitu pada alamat "http://localhost:8100/acl", dengan memberikan data topic yang bernilai "nodeMCU/LED/red/status", acc yang bernilai "1" yang manyatakan bahwa ini merupakan pesan subscribe serta clientid milik klien pertama. Setelah itu diterima respon dari auth-server terkait pesan tersebut, dan dinyatakan bahwa pengguna tersebut tidak terotorisasi untuk menerima / subscribe pesan yang di-publish di topik tersebut, yang dinyatakan dengan "AUTHORIZED=0". Lalu broker tidak akan meneruskan publish tersebut ke klien pertama. Konfigurasi dengan auth-server ini berhasil menolak pesan subscribe yang dikirimkan pada topik yang terdapat pada topic tree namun diluar dari hak akses milik pengguna yang diperiksa pada ACL di database. Dikarenakan pengguna memiliki peran tertentu, dan pada peran tersebut telah dirancang tiap topik yang dapat diakses, maka broker berhasil menolak publish diluar hak akses milik pengguna tersebut. Hasil yang didapatkan oleh broker pada skenario ini dama dengan yang dilakukan pada skenario PMOS 202, karena mekanisme autentikasi yang dibuat pada auth-server tidak diubah.



Gambar 6.45 Screenshot Broker PMOS_302

Seperti yang terlihat pada gambar 6.46, bahwa hasil *sniffing interface* didapatkan tidak ditemukan protokol MQTT, hal ini karena pertukaran paket data dilakukan melalui TLS. Hasil *sniffing* tidak mendapatkan pesan *PUBLISH* yang dikirimkan oleh klien ke broker, namun jika dilihat berdasarkan paket yang dikirim dari ip klien menuju ip broker dapat dilihat pertukaran satu paket *Application*

Data. Application Data ini merupakan pesan PUBLISH yang dikirimkan oleh klien kedua ke broker, namun tidak didapatkan pesan PUBLISH yang diteruskan oleh broker ke klien pertama. Hal ini menandakan bahwa tidak ada pesan yang diteruskan oleh broker ke klien pertama karena ia tidak memiliki hak akses untuk mendapatkan pesan tersebut. Jika dilihat pada isi data dari paket Application Data tersebut dapat dilihat bahwa data yang dikirimkan dapat tidak dapat dibaca karena dalam bentuk enkripsi data.

8		*any											
<u>F</u> ile	<u>E</u> dit	<u>V</u> iew	<u>G</u> o	<u>C</u> apture	<u>A</u> nalyze	<u>S</u> tatistics	Telephony	<u>W</u> ireless	<u>T</u> ools	<u>H</u> elp			
		6		0101 0110 0115	8	۹ 🔶	۵	₹		Ð	Q	€	•
SS SS	sl									×		Exp	ressio
No.	Time	Sourc	e		Destinati	on	Protocol	Length Inf	D				
E 3	3 4.2	192.3	168.1	L00.29	192.168	.100.40	TLSv1.2	135 Ap	plicati	on Data			

Gambar 6.46 Hasil Capture Wireshark PMOS_302

6.2.3.8 Klien Melakukan *Subscribe* Pada Topik Yang Sesuai Dengan Hak Akses Di *Topic Tree* Yang Dibuat. (Kode: PMOS_303)

Pada skenario pengujian ini digunakan konfigurasi broker MQTT yang tidak menerapkan Auth-Server dan menerapkan keamanan TLS. Prosedur pengujian pada skenario ini akan dimulai dengan melakukan *capture interfaces* di Wireshark, lalu dengan aplikasi MQTT-Spy klien pertama akan mengirim pesan *SUBSCRIBE* pada topik yang terdapat dalam *topic tree* namun diluar dari hak akses milik pengguna tersebut ke broker, kemudian klien kedua akan mengirimkan pesan *PUBLISH* yang disesuaikan dengan klien pertama, setelah itu akan dilihat status broker, selanjutnya akan dilakukan *filter* pada hasil *capture interface* menggunakan *field* MQTT yang akan menyeleksi setiap paket yang dikirimkan menggunakan protokol MQTT untuk melihat hasil *sniffing*. Pada pengujian ini digunakan pengguna yang terdaftar dalam sistem yaitu siSuhu yang memiliki peran sebagai temperature_user.

Pada gambar 6.47, dapat dilihat bahwa broker sebelum meneruskan pesan yang di-*publish* oleh klien kedua akan mengirimkan sebuah *request* ke url milik auth-server yang menangani mekanisme otorisasi yaitu pada alamat "http://localhost:8100/acl", dengan memberikan data *topic* yang bernilai "nodeMCU/DHT/temperature/status", acc yang bernilai "1" yang manyatakan bahwa ini merupakan pesan *subscribe* serta clientid milik klien pertama. Setelah itu diterima respon dari auth-server terkait pesan tersebut, dan dinyatakan bahwa pengguna tersebut terotorisasi untuk menerima / *subscribe* pesan yang di-*publish* di topik tersebut, yang dinyatakan dengan "AUTHORIZED=1". Lalu broker akan

meneruskan *publish* tersebut ke klien pertama. Konfigurasi dengan auth-server ini berhasil meneruskan pesan *publish* ke klien pertama yang *subscribe* pada topik yang terdapat pada *topic tree* dan sesuai dengan hak akses milik pengguna tersebut yang diperiksa pada ACL di database. Hasil yang didapatkan oleh broker pada skenario ini dama dengan yang dilakukan pada skenario PMOS_203, karena mekanisme autentikasi yang dibuat pada auth-server tidak diubah.

1509790504: url=http://localhost:8100/acl
1509790504: data=topic=nodeMCU%2FDHT%2Ftemperature%2
Fstatus&acc=1&clientid=Klien_Subscribe
1509790504: aclcheck(eyJhbGciOiJIUzI1NiIsInR5cCI6Ikp
XVCJ9.eyJ1c2VybmFtZSI6InNpU3VodSIsInB3IjoidGVtcDEyMzQifQ
.P4iOiv5hJIbtfIFQ1-nrHqTJJqOPeuBf44NxQ_wu_Dg, nodeMCU/DH
T/temperature/status, 1) trying to acl with jwt
1509790504: aclcheck(eyJhbGciOiJIUzI1NiIsInR5cCI6Ikp
XVCJ9.eyJ1c2VybmFtZSI6InNpU3VodSIsInB3IjoidGVtcDEyMzQifQ
.P4iOiv5hJIbtfIFQ1-nrHqTJJqOPeuBf44NxQ_wu_Dg, nodeMCU/DH
T/temperature/status, 1) AUTHORIZED=1 by jwt

Gambar 6.47 Screenshot Broker PMOS_303

Seperti yang terlihat pada gambar 6.46, bahwa hasil *sniffing interface* didapatkan tidak ditemukan protokol MQTT, hal ini karena pertukaran paket data dilakukan melalui TLS. Hasil *sniffing* tidak mendapatkan pesan *PUBLISH* yang dikirimkan oleh klien ke broker, namun jika dilihat berdasarkan paket yang dikirim dari ip klien menuju ip broker dapat dilihat pertukaran dua paket *Application Data*. Kuda *Application Data* ini merupakan pesan *PUBLISH* yang dikirimkan oleh klien ke broker dapat dilihat pertukaran dua paket *Application Data*. Kuda *Application Data* ini merupakan pesan *PUBLISH* yang dikirimkan oleh klien kedua ke broker dan yang diteruskan oleh broker ke klien pertama. Hal ini menandakan bahwa klien pertama berhasil *subscribe* karena ia memiliki hak akses untuk mendapatkan pesan tersebut. Jika dilihat pada isi data dari paket *Application Data* tersebut dapat dilihat bahwa data yang dikirimkan dapat tidak dapat dibaca karena dalam bentuk enkripsi data.

-	_	_	_	_	_	_		_	_			_	_	_	_	_	_	_		_	
		•	any																		
Ē	ile	<u>E</u> dit	<u>V</u> iev	v <u>G</u> o) <u>C</u> a	apture	<u>A</u> nal	yze	<u>S</u> tat	istics	Tele	ohony	<u>W</u> ire	less	<u>T</u> 00	ls <u>I</u>	<u>H</u> elp				
(5	۲		0101 0110 0111	×	6	9		•	Ď		♪			6	Ð	Q	€,	9
	ssl																Þ	<] -	•	Exp	ressio
No		Time	Sou	rce			Des	tinati	on		Pro	tocol	Lengt	· Info							
	5	15	192	.168	.100.	29	192	.168	.100.	. 40	TLS	v1.2	137	Арр	licat	tion	Data	a			
L C	59	15	192	.168	.100.	.40	192	.168	.100.	.29	TLS	v1.2	137	App	licat	tion	Data	a a			
	72	27	192	.168	.100.	29	192	.168	.100.	. 40	TLS	v1.2	87	Enc	rypte	ed A	lert				
	74	27	192	168	100	29	192	168	100	40	TLS	v1 2	87	Enc	rvnte	Δ he	lert				
•	Fra	ame 59): 13	37 by	tes (on wir	e (10	96 b	its)	, 137	byte	s cap	tured	(109	96 bi	ts)	on i	nter	face	0	
•	Lir	nux co	okec	i cap	ture																
•	Int	ernet	: Pro	otoco	1 Ve	rsion	4, Sr	c: 1	92.1	68.100	9.40,	Dst:	192.1	.68.1	L00.2	9					
•	Tra	ansmis	sior	n Con	trol	Proto	col,	Src	Port	: 8883	3, Ds [.]	t Por	t: 621	L63,	Seq:	1,	Ack:	1,	Len:	81	
•	Sec	cure S	ocke	ets L	ayer																
	•	TLSv1	.2 R	ecor	d Lay	/er: A	pplic	atio	n Dat	ta Pro	tocol	L: mqt	tt								
		Co	nten	t Typ	be: A	Applica	ation	Data	a (23	3)											
1		Ve	rsio	n: Ti	S 1.	2 (0x0	9303)														
1	Length: 76																				
•		En	cryp	ted A	Appli	catio	n Data	a: 83	319c4	47c1f1	8bf2a	c1426	6ca087	3a13	53573	3b64(6886a	te16	3a		

Gambar 6.48 Hasil Capture Wireshark PMOS_303

6.3 Pengujian Performa Sistem

Pengujian performa sistem dilakukan untuk membandingkan performa sistem berdasarkan pada ketiga konfigurasi broker yang dilakukan dalam pengujian keamanan sistem. Pengujian performa akan difokuskan untuk menguji parameter waktu yang dibutuhkan untuk melakukan koneksi dengan broker, waktu yang dibutuhkan untuk melakukan *publish* pesan dan *throughput* yaitu jumlah pesan yang mampu ditangani setiap satu detik. Agar pengujian dapat difokuskan pada parameter yang diuji, maka ada beberapa parameter eksternal yang diabaikan, seperti besarnya data *payload* yang dikirimkan, *Quality of Services* pada MQTT, interval pengiriman tiap paket, dan diasumsikan bahwa tidak ada kegagalan dalam pengiriman paket. Pengujian ini diharapkan di setiap parameter dapat memenuhi hasil yang diharapkan pada tabel 6.2.

Kode	Parameter	Hasil yang
Roue	Falameter	diharapkan
PPS_001	Waktu yang dibutuhkan untuk membangun koneksi (Average Connection Time)	< 0.5 detik
PPS_002	Waktu yang dibutuhkan untuk pengiriman / publish satu pesan (Average Publish Time)	< 1 detik
PPS_003	Jumlah pesan yang mampu dikirimkan dalam satu detik (<i>Throughput</i>)	> 100 pesan/detik

Tabel 6.2 Paramater Pengujian Performa Sistem

Pengujian akan dilakukan dengan menggunakan sebuah *tools* pada node.js bernama MQTT-*Benchmark* agar dapat mendukung jumlah klien MQTT yang akan digunakan. Pada setiap skenario akan digunakan 10 klien MQTT yang akan melakukan koneksi ke broker, kemudian setiap klien MQTT akan melakukan *publish* pesan sebanyak 100 kali. Kemudian MQTT-*Benchmark* akan melakukan kalkulasi untuk mendapatkan setiap parameter pengujian performa yang sudah dijelaskan. Pada pengujian ini digunakan pengguna yang terdaftar dalam sistem yaitu siMerah yang memiliki peran sebagai red_user.



Gambar 6.49 Grafik Hasil Pengujian PPS_001

Dari hasil pengujian performa PPS_001 yang digambarkan pada grafik pada gambar 6.49, waktu yang dibutuhkan oleh klien MQTT untuk melakukan koneksi ke broker didapatkan bahwa jumlah waktu yang dibutuhkan oleh konfigurasi broker tanpa auth-server dan tanpa tls sebesar 0,0288 detik. Pada konfigurasi dengan auth-server namun tanpa TLS sebesar 0,0662 detik dan pada konfigurasi dengan auth-server dan TLS sebesar 0,0853 detik. Hasil ini berbanding lurus dengan kompleksitas pemeriksaan autentikasi dan pemeriksaan sertifikat di TLS, namun berdasarkan hasil yang didapatkan telah memenuhi hasil yang diharapkan, bahwa klien berhasil melakukan koneksi pada waktu dibawah 0,5000 detik.



Gambar 6.50 Grafik Hasil Pengujian PPS_002

Dari hasil pengujian performa PPS_002 yang digambarkan pada grafik pada gambar 6.50, waktu yang dibutuhkan oleh klien MQTT untuk melakukan publish ke broker didapatkan bahwa jumlah waktu yang dibutuhkan oleh konfigurasi broker tanpa auth-server dan tanpa tls sebesar 0,1379 detik. Pada konfigurasi dengan auth-server namun tanpa TLS sebesar 0,4883 detik dan pada konfigurasi dengan auth-server dan TLS sebesar 0,7092 detik. Hasil ini berbanding lurus dengan kompleksitas pemeriksaan otorisasi dan pemeriksaan sertifikat di TLS, namun berdasarkan hasil yang didapatkan telah memenuhi hasil yang diharapkan, bahwa klien berhasil melakukan *publish* pada waktu dibawah 1 detik. Jika dibandingkan dengan waktu yang dibutuhkan untuk melakukan koneksi waktu yang dibutuhkan untuk melakukan lebih lama dikarenakan data payload yang ditransmisikan lebih besar.



Gambar 6.51 Grafik Hasil Pengujian PPS_003

Dari hasil pengujian performa PPS_002 yang digambarkan pada grafik pada gambar 6.51, jumlah pesan yang berhasil dikirimkan pada satu detik didapatkan bahwa jumlah pesan yang dikirimkan ke konfigurasi broker tanpa auth-server dan tanpa tls sebesar 925 pesan per detik. Pada konfigurasi dengan auth-server namun tanpa TLS sebesar 205 pesa per detik dan pada konfigurasi dengan auth-server dan TLS sebesar 141 pesan per detik. Hasil ini berbanding lurus dengan kompleksitas pemeriksaan otorisasi dan pemeriksaan sertifikat di TLS, namun berdasarkan hasil yang didapatkan telah memenuhi hasil yang diharapkan, bahwa klien berhasil melakukan *publish* sebanyak 100 pesan per detik.

Berdasarkan pada hasil pengujian yang di dapatkan tersebut, diketahui bahwa konfigurasi broker yang dibuat mempengaruhi jumlah waktu yang dibutuhkan untuk melakukan koneksi, waktu yang dibutuhkan untuk *publish* pesan dan jumlah pesan yang mampu ditangani oleh server dalam satu detik. Namun perbedaan diantara kedua konfigurasi menggunakan auth-server tanpa TLS dan auth-server yang menggunakan TLS tidak terlalu signifikan dan masih memenuhi hasil yang diharapkan. Tetapi jika dibandingkan dengan konfigurasi *default* sistem kedua konfigurasi tersebut cukup signifikan, namun dalam faktor keamanan yang ada di sistem maka konfigurasi ini tidak direkomendasikan untuk digunakan.