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.

Tabel 6.1 Hasil Pengujian Fungsional Sistem

Kode	Fungsi	Hasil
PFS_001	Klien MQTT dapat membangun koneksi dengan broker MQTT menggunakan token JWT	Berhasil
PFS_002	*Publisher* 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 *generate* token JWT berdasakan *username* dan *password* pengguna yang ada di database	Berhasil
PFS_007	Admin dapat melakukan *Create User* untuk membuat sebuah pengguna baru lewat auth-server	Berhasil
PFS_008	Admin dapat melakukan *Update User* untuk mengubah pengguna lewat auth-server	Berhasil
PFS_009	Admin dapat melakukan *Delete User* untuk menghapus pengguna lewat auth-server	Berhasil
PFS_010	Admin dapat melakukan *Create Role* untuk membuat peran baru lewat auth-server	Berhasil

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

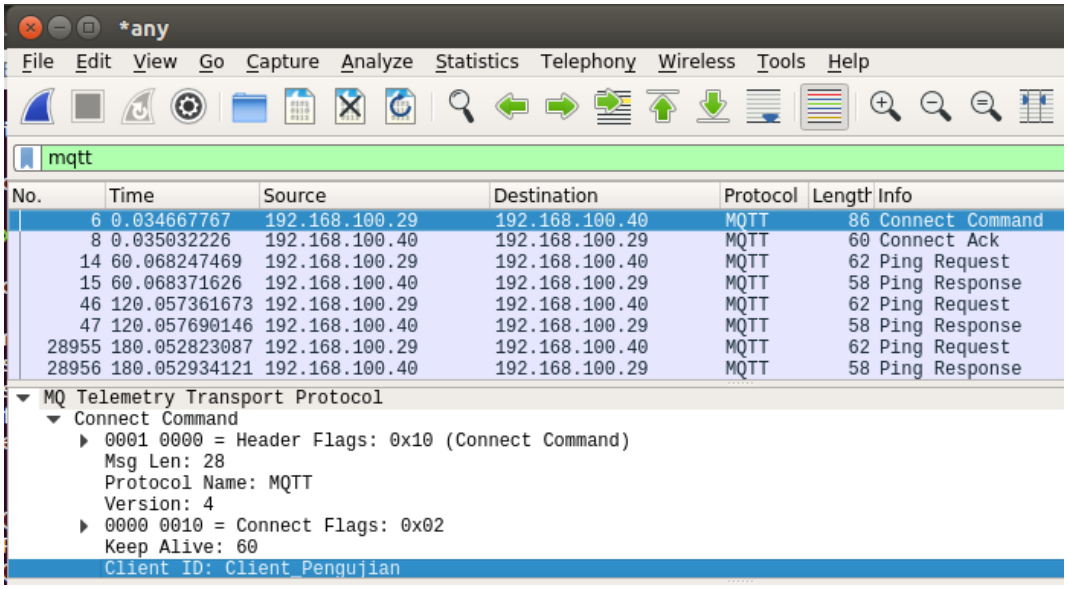
6.2.1.1 Klien Membangun Koneksi Dengan Broker Tanpa Menggunakan *Username* Dan *Password*. (Kode: PMO_101)

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 on port 1883.
1509616223: New client connected 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*.

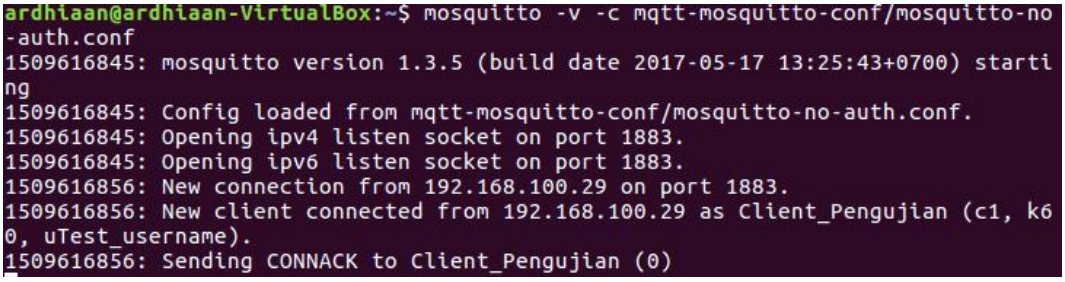


Gambar 6.2 Hasil Capture Wireshark PMO_101

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)

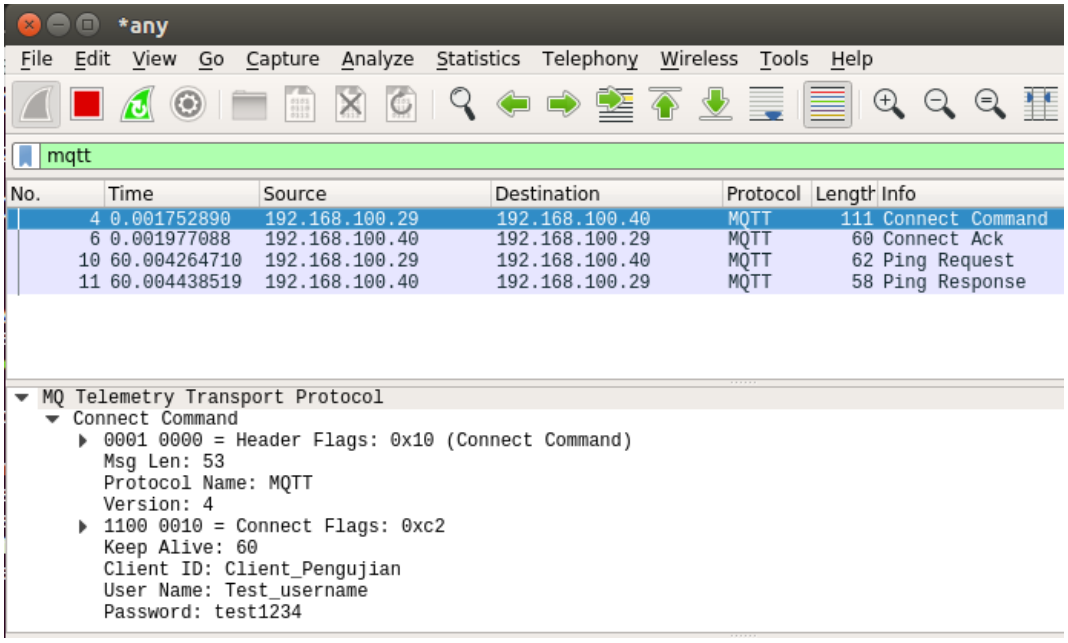
Pada skenario pengujian ini digunakan konfigurasi broker MQTT yang tidak menerapkan Auth-Server dan tidak menerapkan TLS. Prosedur pengujian pada skenario ini akan dimulai dengan melakukan *capture interfaces* di Wireshark, lalu dengan aplikasi MQTT-Spy klien akan mengirim pesan *CONNECT* ke broker dengan 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*.



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".



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*.

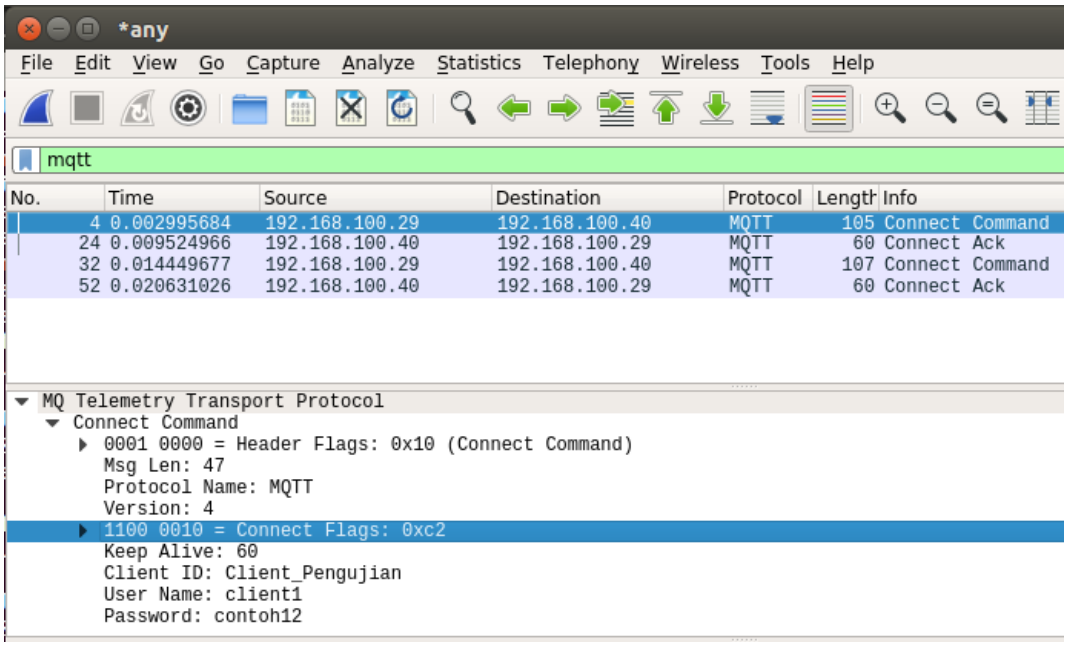
```
1509617711: New connection from 192.168.100.29 on port 1883.
1509617711: |-- mosquitto_auth_unpwd_check(client1)
1509617711: |-- ** checking backend jwt
1509617711: |-- url=http://localhost:8100/auth
1509617711: |-- data=topic=&acc=-1&clientid=
1509617711: |-- getuser(client1) AUTHENTICATED=0 by none
1509617711: Sending 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".



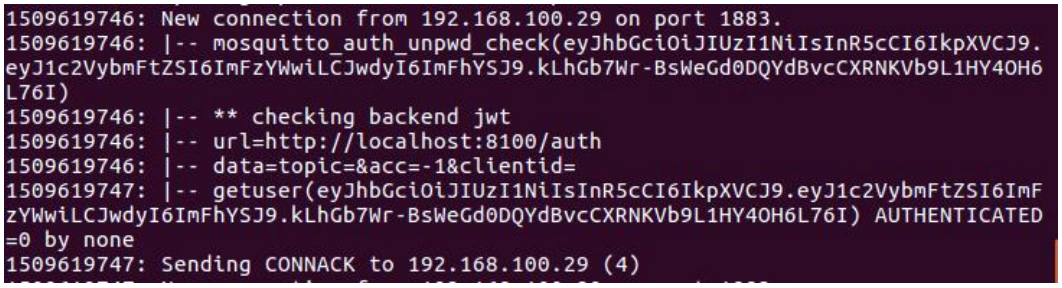
Gambar 6.6 Hasil Capture Wireshark PMO_201

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*.

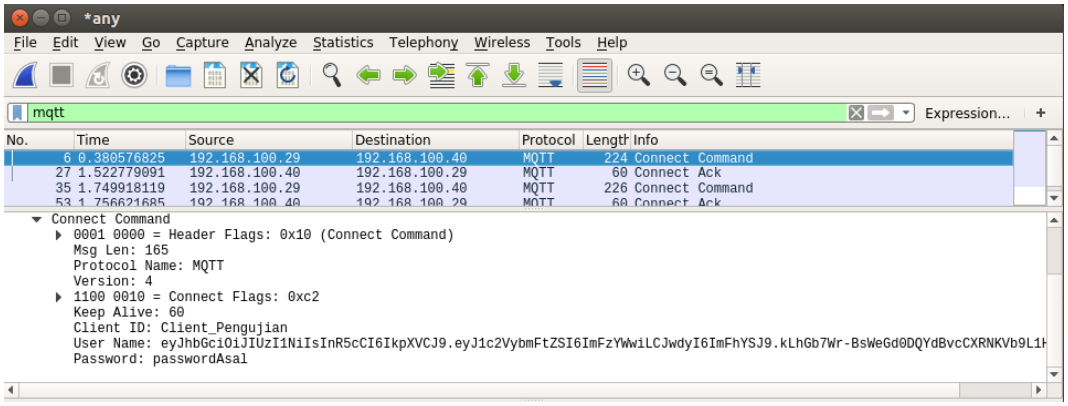
Pada gambar 6.7, dapat dilihat bahwa broker berhasil menolak permintaan koneksi dari klien dan mengirimkan pesan CONNACK bernilai 4, yang menandakan permintaan koneksi ditolak karena tidak terautentikasi, dengan menggunakan

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.

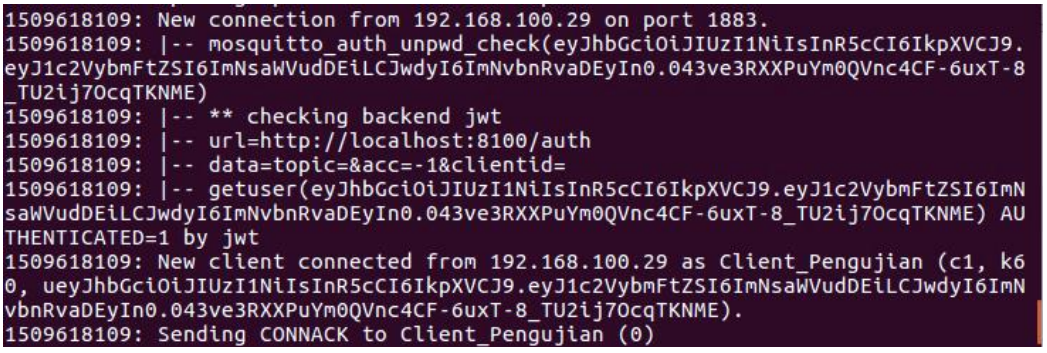


Gambar 6.8 Hasil Capture Wireshark PMO_202

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*.

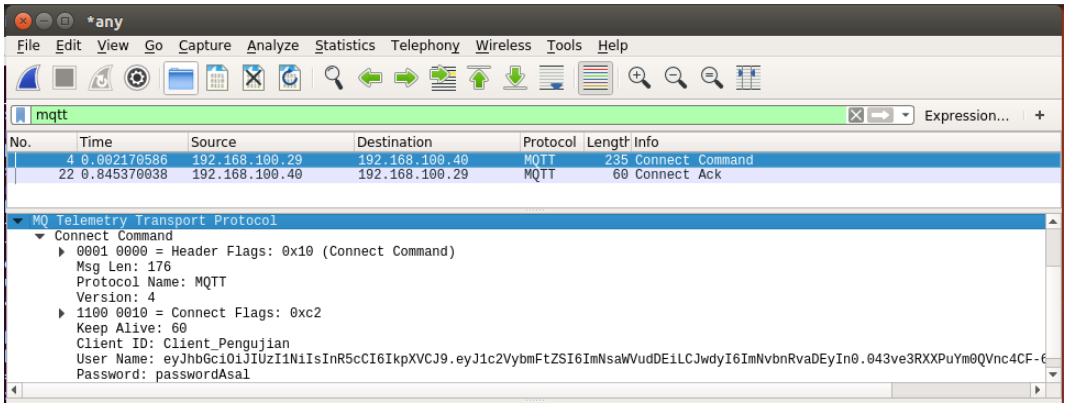
Pada gambar 6.9, dapat dilihat bahwa broker berhasil menerima permintaan koneksi dari klien dan mengirimkan pesan CONNACK bernilai 0, yang menandakan permintaan koneksi diterima, dengan menggunakan konfigurasi ini klien dapat melakukan koneksi dengan broker karena menggunakan format *username* dengan 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.



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

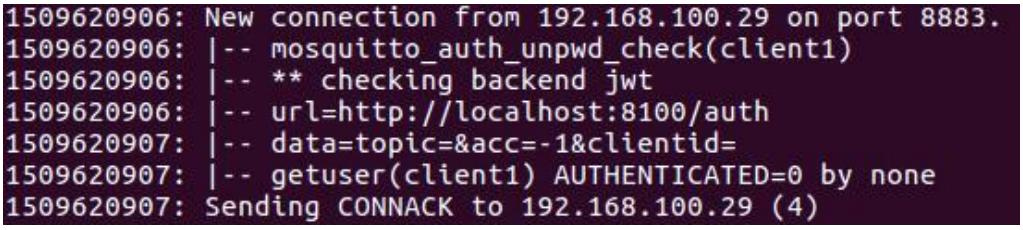
diambil nilainya dan dapat digunakan untuk mengakses sistem sebagai klien ini, sehingga masih dibutuhkan faktor keamanan tambahan. Hasil yang didapatkan dari *sniffing* paket MQTT pada skenario ini sama dengan yang ada pada skenario PMO_202.



Gambar 6.10 Hasil Capture Wireshark PMO_203

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

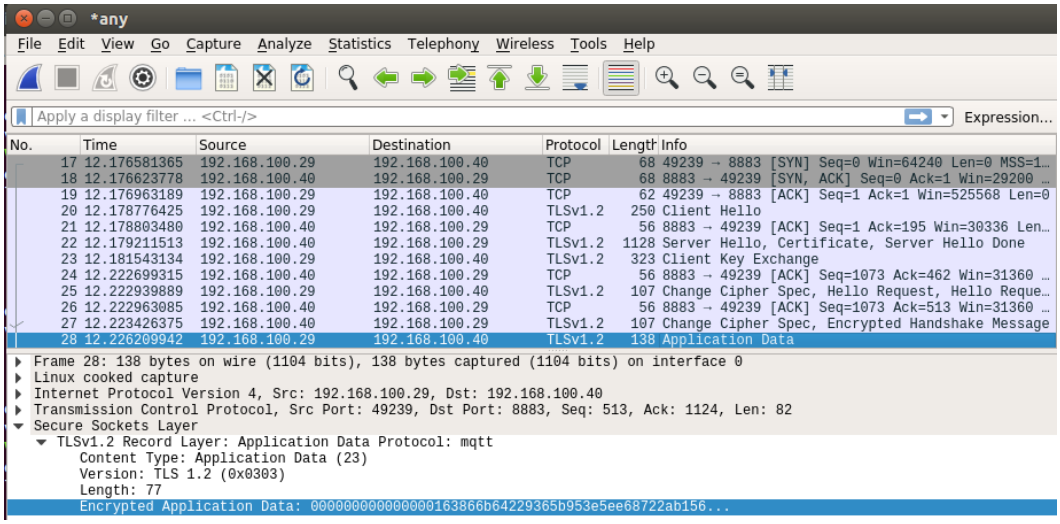
Pada skenario pengujian ini digunakan konfigurasi broker MQTT yang akan menerapkan Auth-Server menggunakan *backend* JWT dan menerapkan keamanan TLS. Prosedur pengujian pada skenario ini akan dimulai dengan melakukan *capture interfaces* di Wireshark, lalu dengan aplikasi MQTT-Spy klien akan mengirim 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.



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.



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.

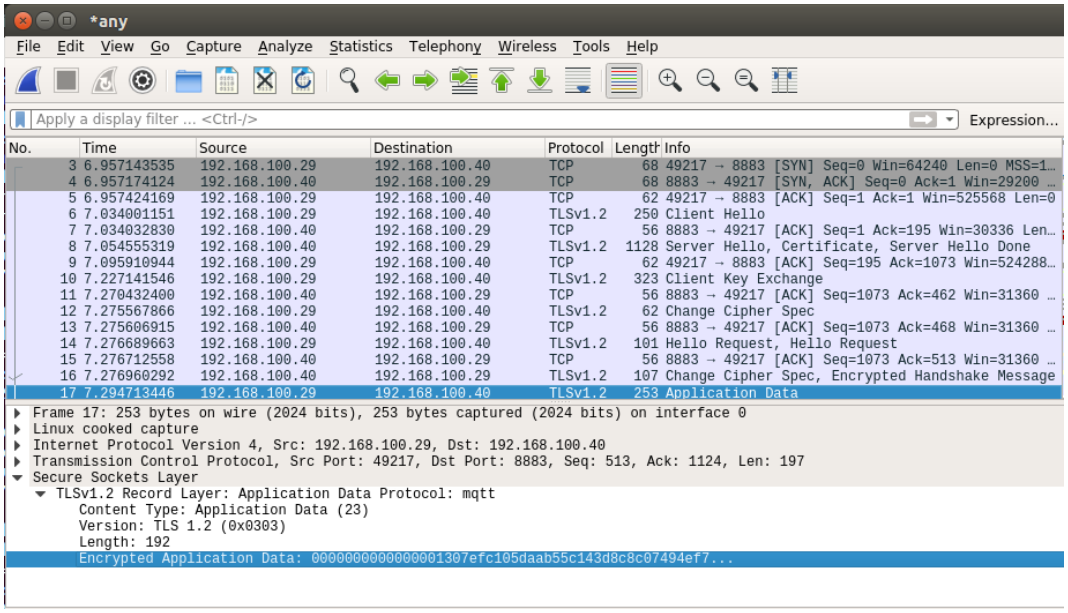
6.2.1.7 Klien Membangun Koneksi Dengan Broker Dengan Menggunakan Token JWT yang *Invalid*. (Kode: PMO_302)

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.
eyJ1c2VybmFtZSI6ImFzYWwiLCJwdyI6ImFhYSJ9.kLhGb7Wr-BsWeGd0DQYdBvcCXRNKVb9L1HY4OH6
L76I)
1509620245: |-- ** checking backend jwt
1509620245: |-- url=http://localhost:8100/auth
1509620245: |-- data=topic=&acc=-1&clientid=
1509620245: |-- getuser(eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmFtZSI6ImF
zYWwiLCJwdyI6ImFhYSJ9.kLhGb7Wr-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.

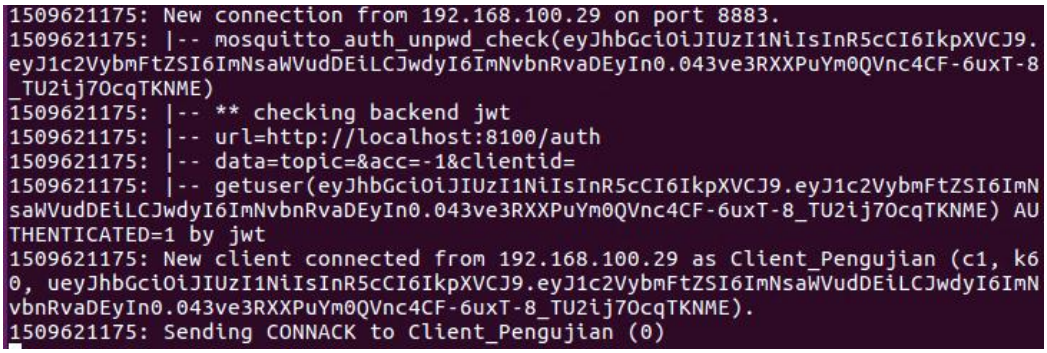


Gambar 6.14 Hasil Capture Wireshark PMO_302

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)

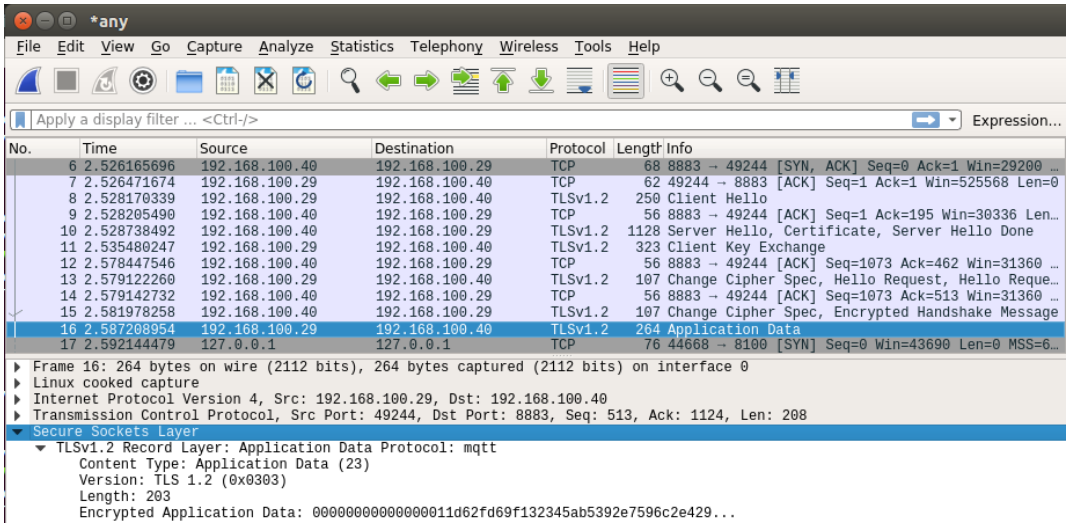
Pada skenario pengujian ini digunakan konfigurasi broker MQTT yang akan menerapkan Auth-Server menggunakan *backend* JWT dan menerapkan keamanan TLS. Prosedur pengujian pada skenario ini akan dimulai dengan melakukan *capture interfaces* di Wireshark, lalu dengan aplikasi MQTT-Spy klien akan mengirim 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.



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 auth-server tidak diubah.



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.

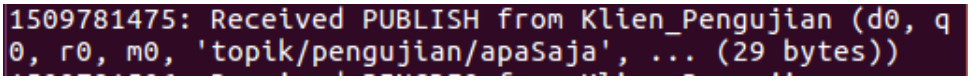
6.2.2 Pengujian Mekanisme Otorisasi Sistem Melalui Publish

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.

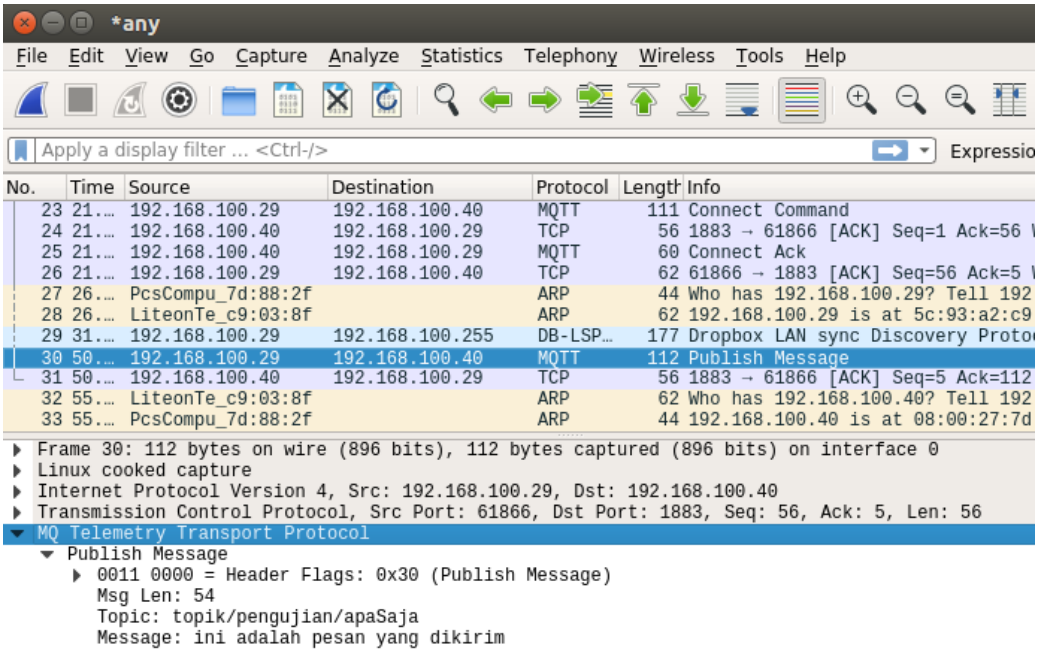
6.2.2.1 Klien Melakukan *Publish* Pada Topik Yang Tidak Ada Dalam *Topic Tree* Yang Dibuat. (Kode: PMOP_101)

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.



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".

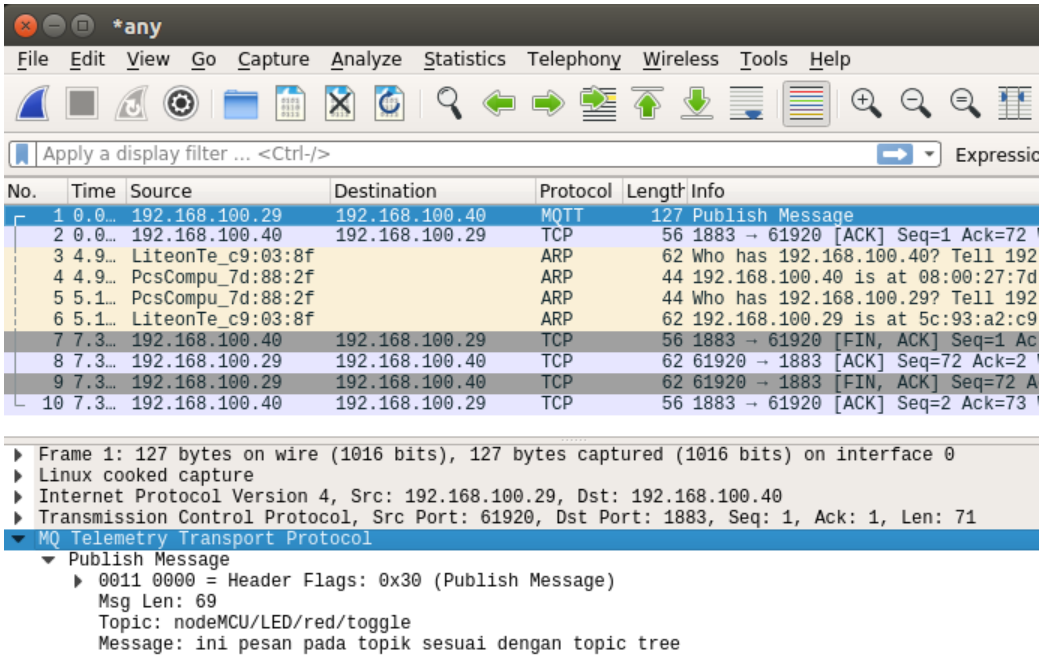
6.2.2.2 Klien Melakukan *Publish* Pada Topik Yang Ada Dalam *Topic Tree* Yang Dibuat. (Kode: PMOP_102)

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*.

```
1509782066: Received PUBLISH from Klien_Pengujian (d0, q
0, r0, m0, 'nodeMCU/LED/red/toggle', ... (45 bytes))
```

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.



Gambar 6.20 Hasil Capture Wireshark PMOP_102

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.

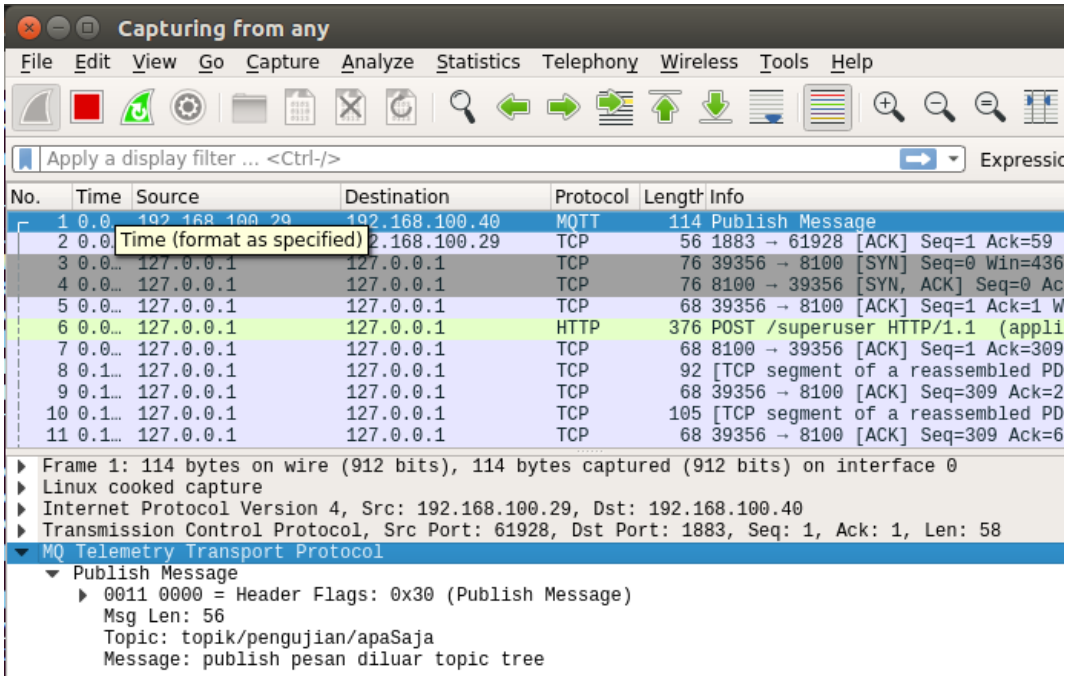
6.2.2.3 Klien Melakukan *Publish* Pada Topik Diluar *Topic Tree* Yang Dibuat. (Kode: PMOP_201)

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,eyJhbGciOiJIUzI1NiIsInR
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.



Gambar 6.22 Hasil Capture Wireshark PMOP_201

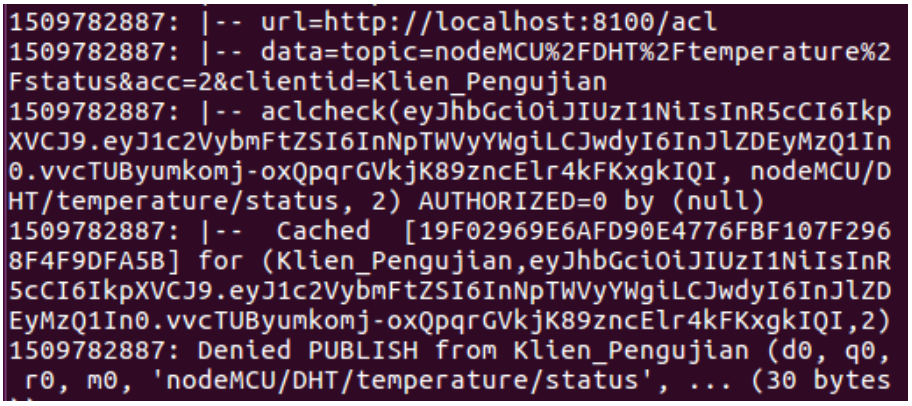
Seperti yang terlihat pada gambar 6.22, bahwa hasil *sniffing* protokol MQTT mendapatkan pesan *PUBLISH* yang dikirimkan oleh klien ke broker. Jika dilihat pada isi data dari paket *publish* tersebut dapat dilihat bahwa data yang dikirimkan dapat dibaca dalam bentuk *plain text*, kemudian jika dilihat 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.

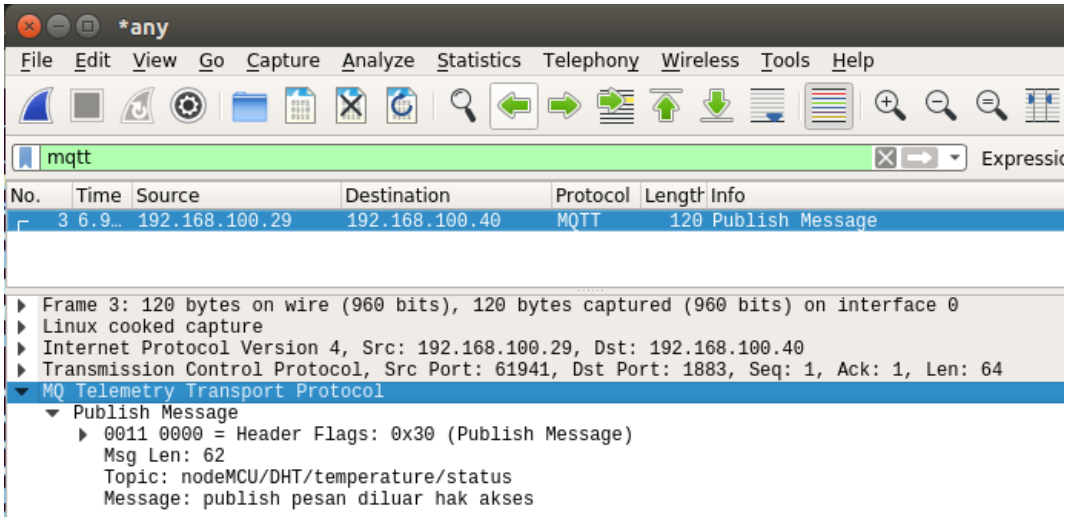
Pada gambar 6.23, dapat dilihat bahwa broker menerima sebuah pesan *PUBLISH* kemudian mengirimkan pesan tersebut ke url milik auth-server yang menangani mekanisme otorisasi yaitu "http://localhost:8100/acl", dengan data *topic* yang bernilai "nodeMCU/DHT/temperature/status", acc yang bernilai "2" yang 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.



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".

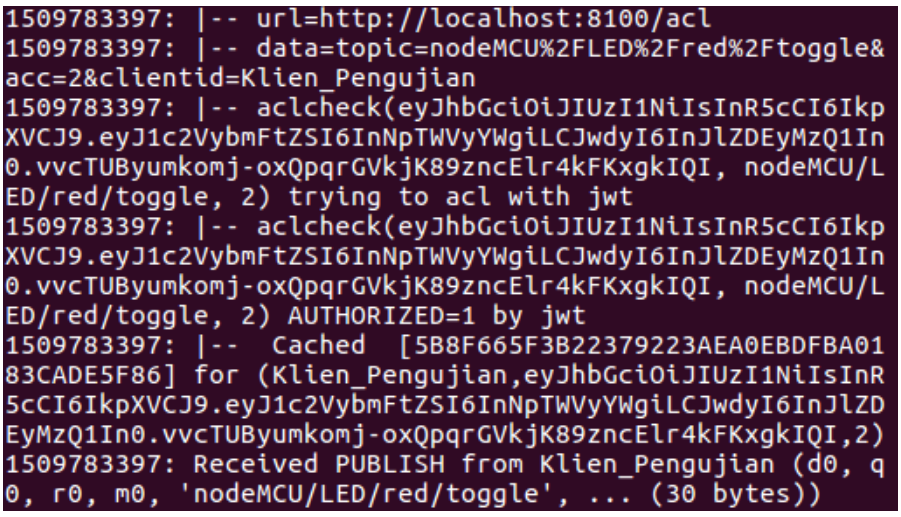


Gambar 6.24 Hasil Capture Wireshark PMOP_202

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.

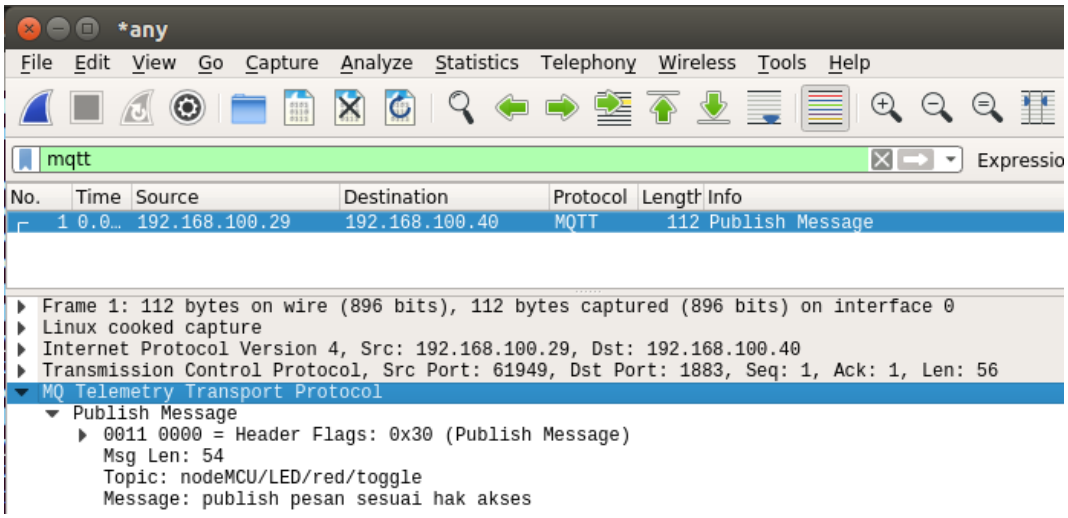
Pada gambar 6.25, dapat dilihat bahwa broker menerima sebuah pesan *PUBLISH* kemudian mengirimkan pesan tersebut ke url milik auth-server yang menangani mekanisme otorisasi yaitu "http://localhost:8100/acl", dengan data *topic* yang bernilai "nodeMCU/LED/red/toggle", acc yang bernilai "2" yang manyatakan 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.



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.

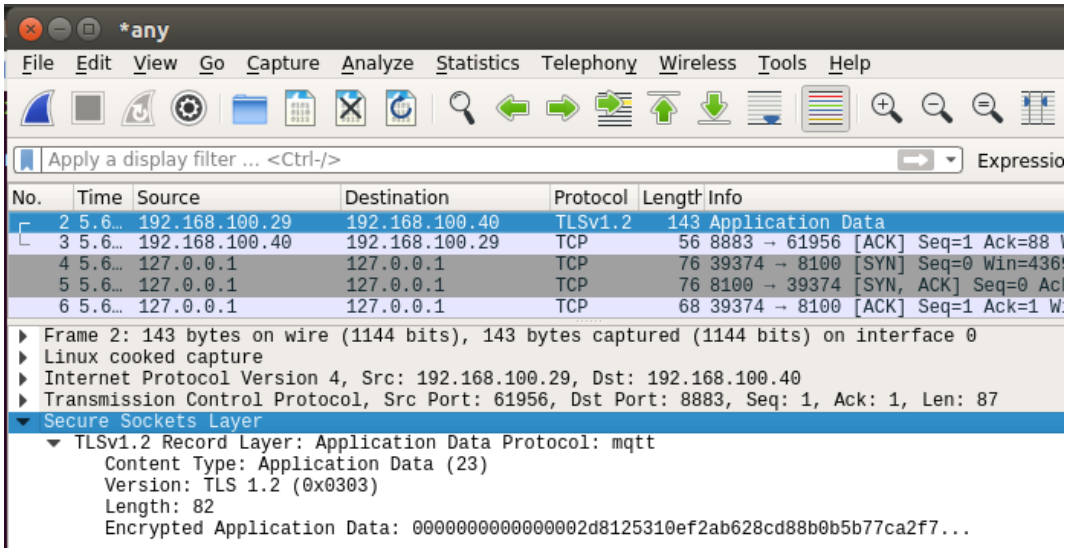
Pada gambar 6.27, dapat dilihat bahwa broker menerima sebuah pesan *PUBLISH* kemudian mengirimkan pesan tersebut ke url milik auth-server yang menangani mekanisme otorisasi yaitu "http://localhost:8100/acl", dengan data *topic* yang bernilai "topik/pengujian/apaSaja", acc yang bernilai "2" yang manyatakan 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(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
EyMzQ1In0.vvcTUByumkomj-oxQpqrGVkjK89zncElr4kFKxgkIQI,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.



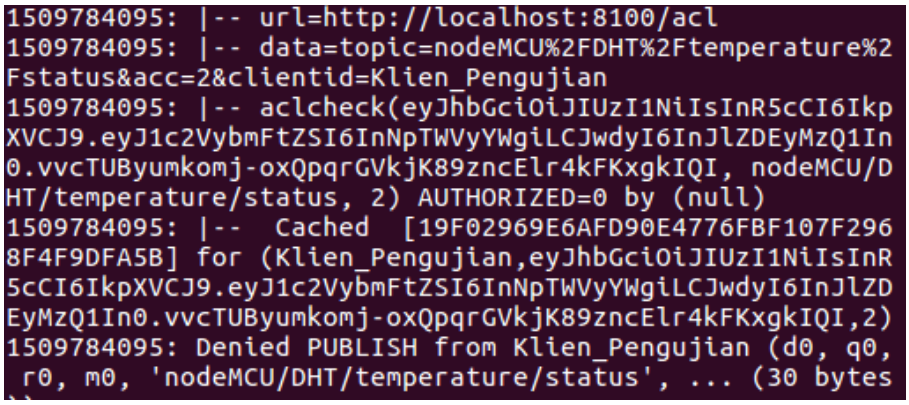
Gambar 6.28 Hasil Capture Wireshark PMOP_301

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

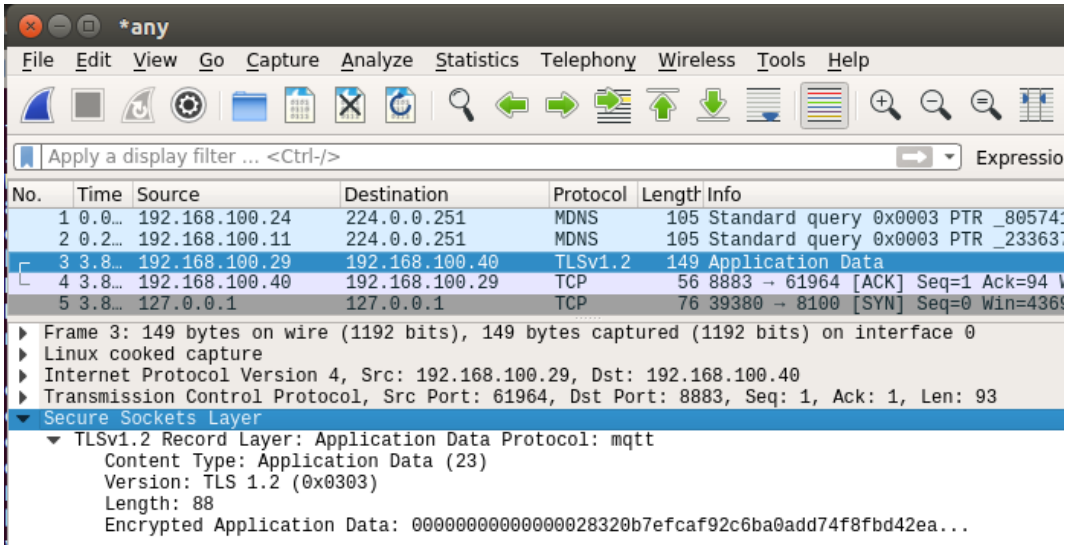
milik pengguna tersebut ke broker, setelah itu akan dilihat status broker, selanjutnya akan dilakukan *filter* pada hasil *capture interface* menggunakan *field* MQTT yang akan menyeleksi setiap paket yang dikirimkan menggunakan protokol MQTT untuk melihat hasil *sniffing*. Pada pengujian ini digunakan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



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)

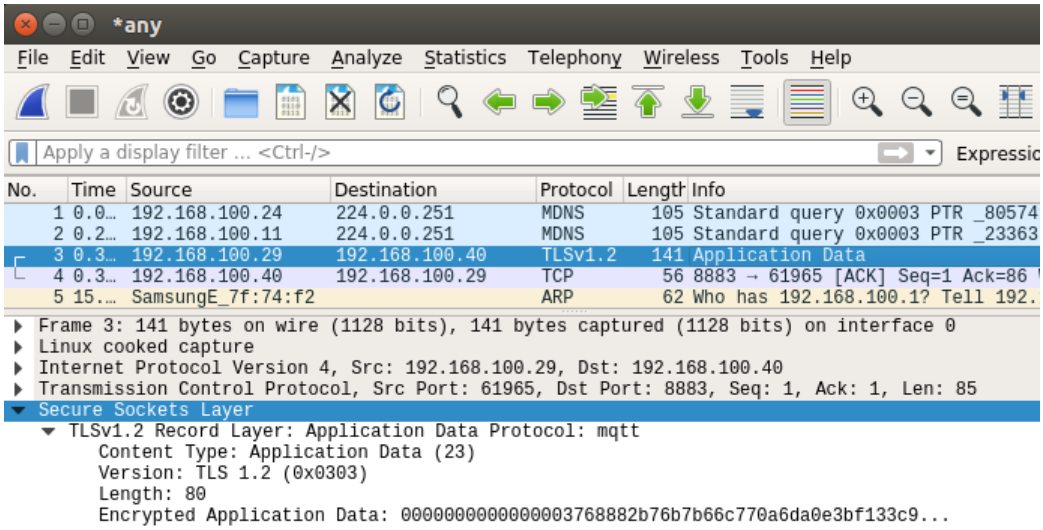
Pada skenario pengujian ini digunakan konfigurasi broker MQTT yang akan menerapkan Auth-Server menggunakan *backend* JWT dan menerapkan keamanan TLS. Prosedur pengujian pada skenario ini akan dimulai dengan melakukan *capture interfaces* di Wireshark lalu dengan aplikasi MQTT-Spy klien akan mengirim 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.



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 pertama berhasil menerima pesan tersebut, hasil *capture* dari Wireshark dan 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*.

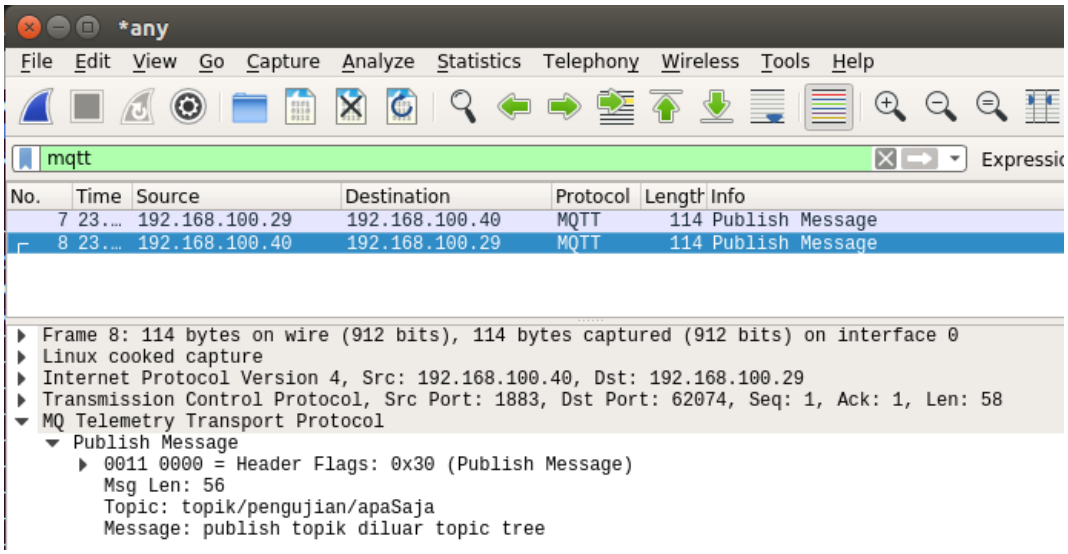
Pada gambar 6.33, dapat dilihat bahwa broker meneruskan pesan *PUBLISH* dari klien kedua ke klien pertama, dengan menggunakan konfigurasi ini klien pertama dapat *subscribe* pada topik manapun baik yang dirancang pada *topic tree* maupun diluar dari rancangan, sehingga dibutuhkan sebuah mekanisme otorisasi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

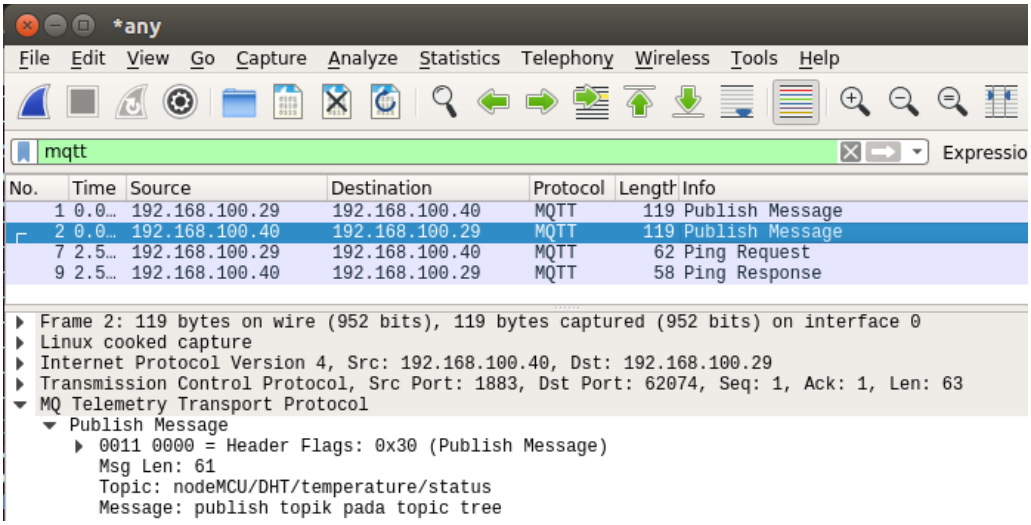
Pada gambar 6.35, dapat dilihat bahwa broker meneruskan pesan *PUBLISH* dari klien kedua ke klien pertama, dengan menggunakan konfigurasi ini klien pertama dapat *subscribe* pada topik manapun baik yang dirancang pada *topic tree* maupun diluar dari rancangan, sehingga dibutuhkan sebuah mekanisme otorisasi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



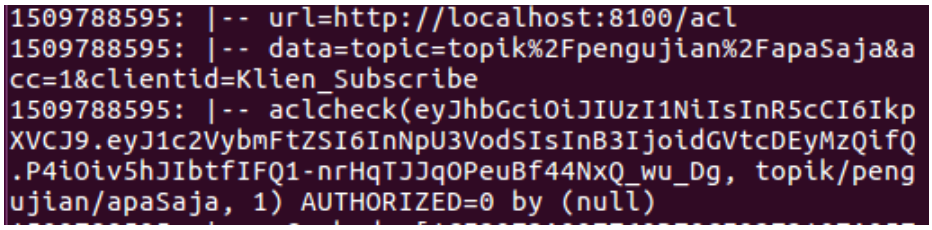
Gambar 6.36 Hasil Capture Wireshark PMOS_102

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.

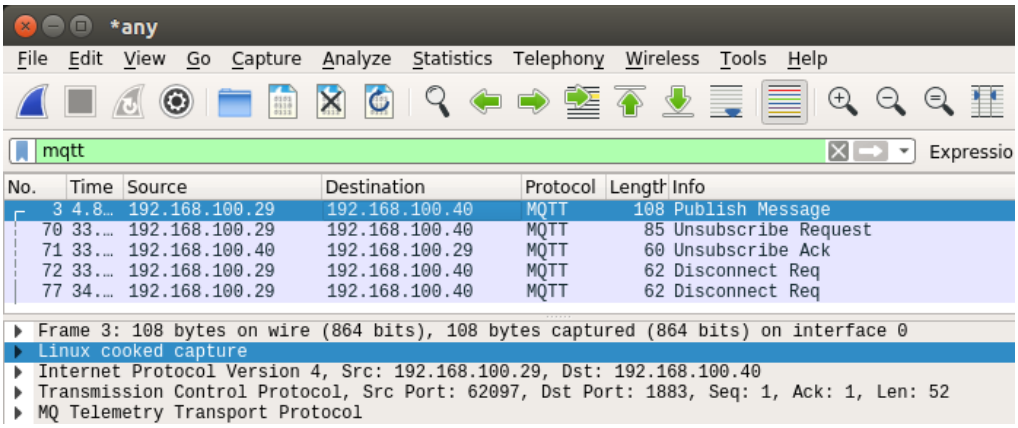
Pada gambar 6.37, dapat dilihat bahwa broker sebelum meneruskan pesan yang di-*publish* oleh klien kedua akan mengirimkan sebuah *request* ke url milik auth-server yang menangani mekanisme otorisasi yaitu pada alamat "http://localhost:8100/acl", dengan memeberikan data *topic* yang bernilai "topik/pengujian/apaSaja", 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.

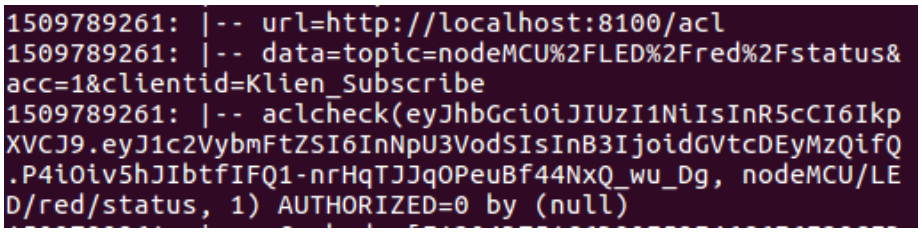


Gambar 6.38 Hasil Capture Wireshark PMOS_201

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

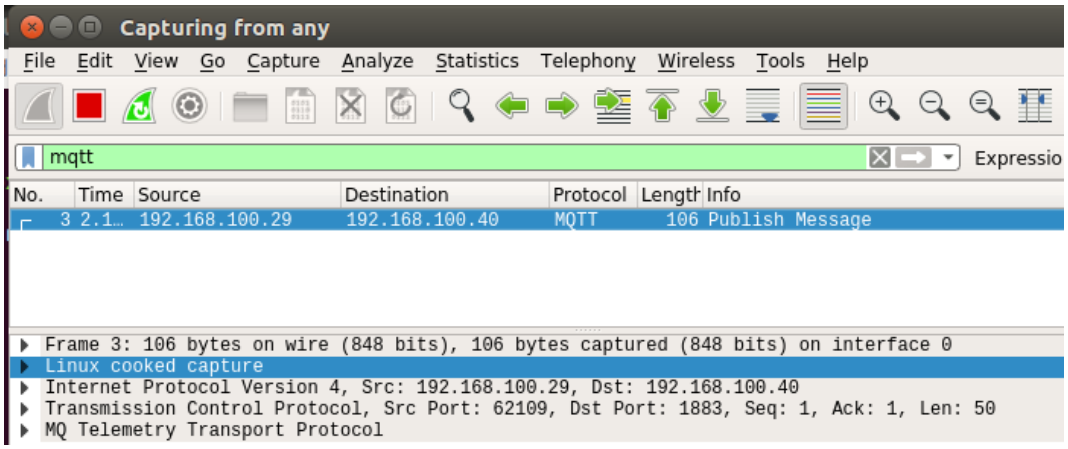
Pada skenario pengujian ini digunakan konfigurasi broker MQTT yang tidak menerapkan Auth-Server dan tidak menerapkan TLS. Prosedur pengujian pada skenario ini akan dimulai dengan melakukan *capture interfaces* di Wireshark, lalu dengan aplikasi MQTT-Spy klien pertama akan mengirim pesan *SUBSCRIBE* pada 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.

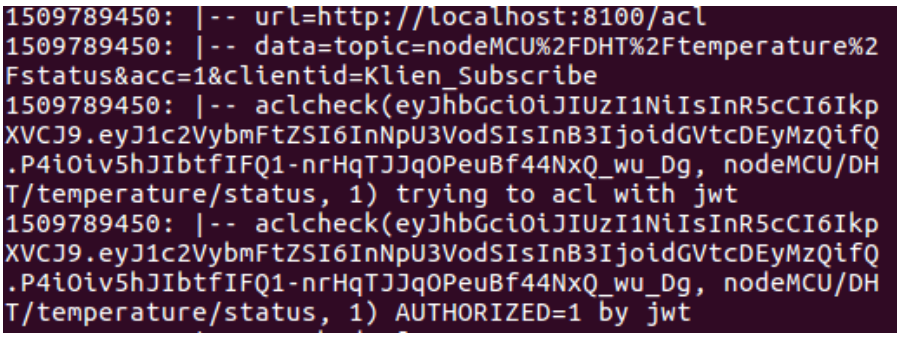


Gambar 6.40 Hasil Capture Wireshark PMOS_202

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.

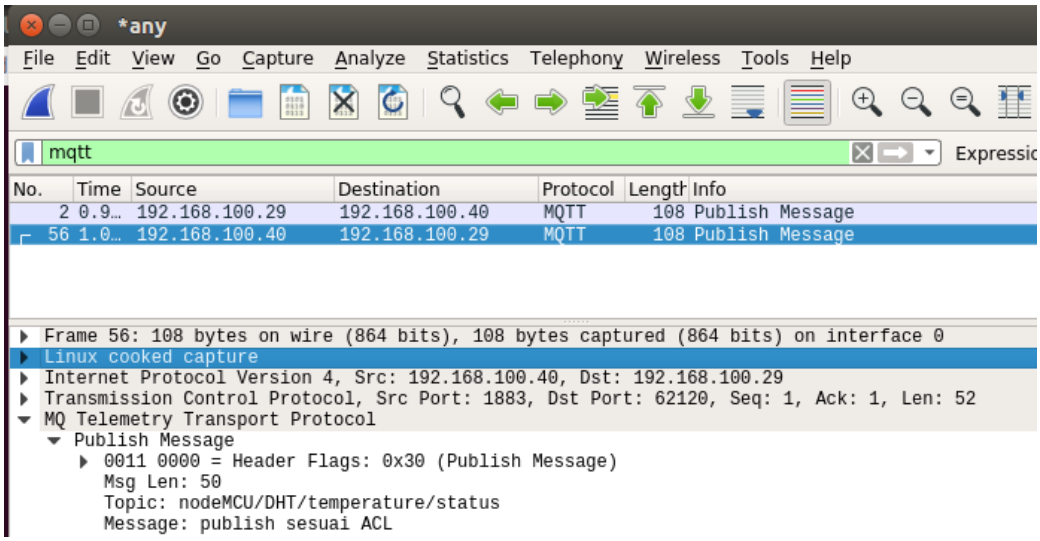
Pada gambar 6.41, dapat dilihat bahwa broker sebelum meneruskan pesan yang di-*publish* oleh klien kedua akan mengirimkan sebuah *request* ke url milik auth-server yang menangani mekanisme otorisasi yaitu pada alamat "http://localhost:8100/acl", dengan memberikan data *topic* yang bernilai "nodeMCU/DHT/temperature/status", 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.

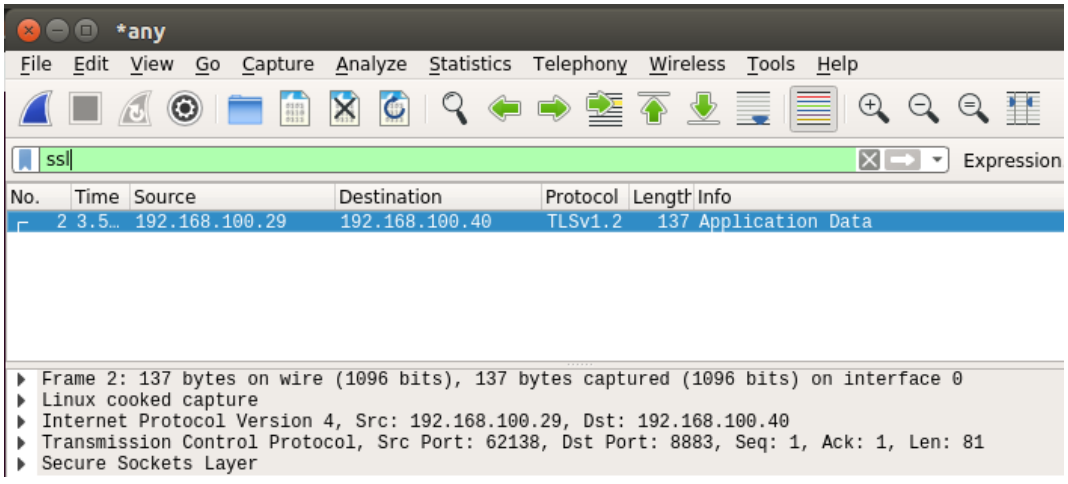
Pada gambar 6.43, dapat dilihat bahwa broker sebelum meneruskan pesan yang di-*publish* oleh klien kedua akan mengirimkan sebuah *request* ke url milik auth-server yang menangani mekanisme otorisasi yaitu pada alamat "http://localhost:8100/acl", dengan memeberikan data *topic* yang bernilai "topik/pengujian/apaSaja", 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.eyJ1c2VybmFtZSI6InNpU3VodSIsInB3IjoidGVtcDEyMzQifQ
.P4iOiv5hJIbtfIFQ1-nrHqTJJqOPeuBf44NxQ_wu_Dg, topik/peng
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*. *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.

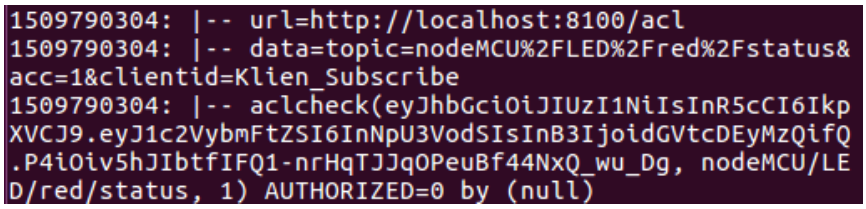


Gambar 6.44 Hasil Capture Wireshark PMOS_301

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.

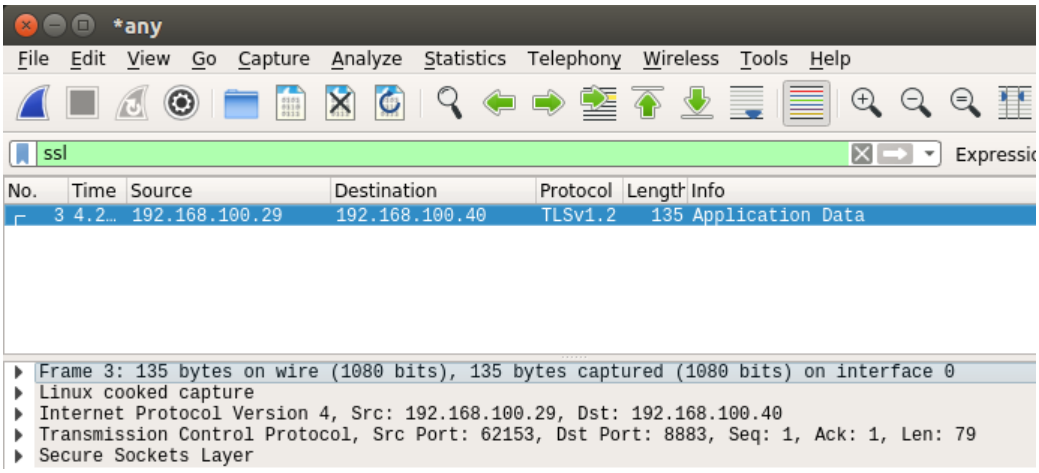
Pada gambar 6.45, dapat dilihat bahwa broker sebelum meneruskan pesan yang di-*publish* oleh klien kedua akan mengirimkan sebuah *request* ke url milik auth-server yang menangani mekanisme otorisasi yaitu pada alamat "http://localhost:8100/acl", dengan memberikan data *topic* yang bernilai "nodeMCU/LED/red/status", 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.



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.

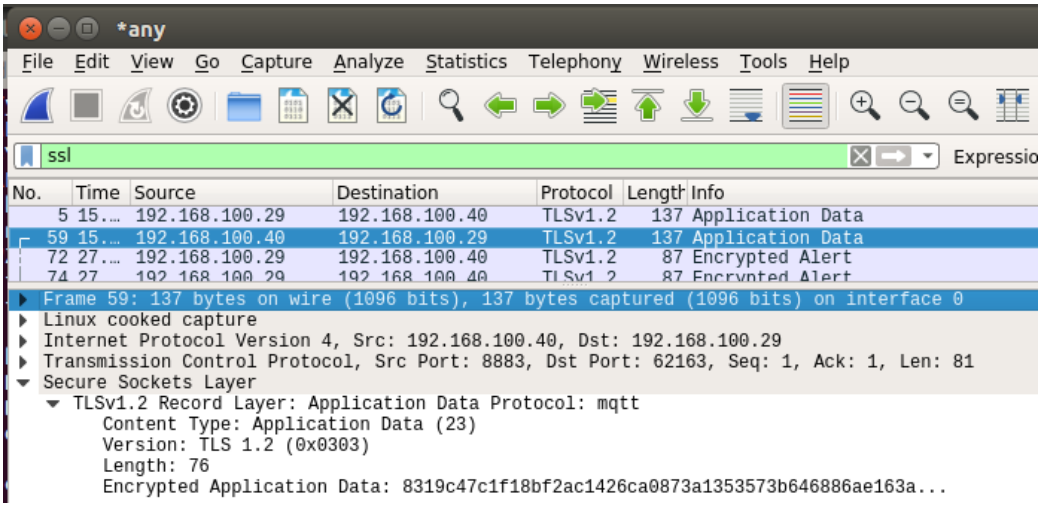
Pada gambar 6.47, dapat dilihat bahwa broker sebelum meneruskan pesan yang di-*publish* oleh klien kedua akan mengirimkan sebuah *request* ke url milik auth-server yang menangani mekanisme otorisasi yaitu pada alamat "http://localhost:8100/acl", dengan memberikan data *topic* yang bernilai "nodeMCU/DHT/temperature/status", 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 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.



Gambar 6.48 Hasil Capture Wireshark PMOS_303

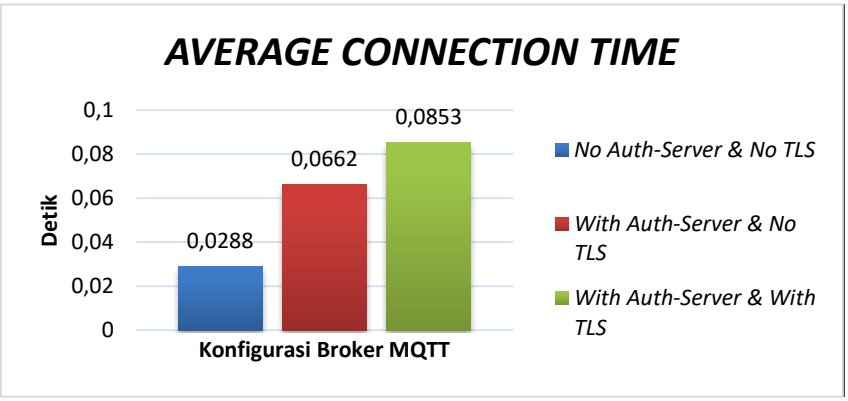
6.3 Pengujian Performa Sistem

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.

Tabel 6.2 Paramater Pengujian Performa Sistem

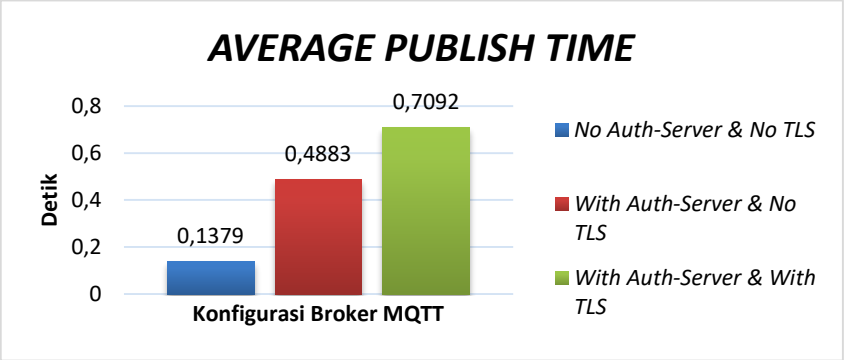
Kode	Parameter	Hasil yang diharapkan
PPS_001	Waktu yang dibutuhkan untuk membangun koneksi (*Average Connection Time*)	< 0.5 detik
PPS_002	Waktu yang dibutuhkan untuk pengiriman / *publish* satu pesan (*Average Publish Time*)	< 1 detik
PPS_003	Jumlah pesan yang mampu dikirimkan dalam satu detik (*Throughput*)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

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.



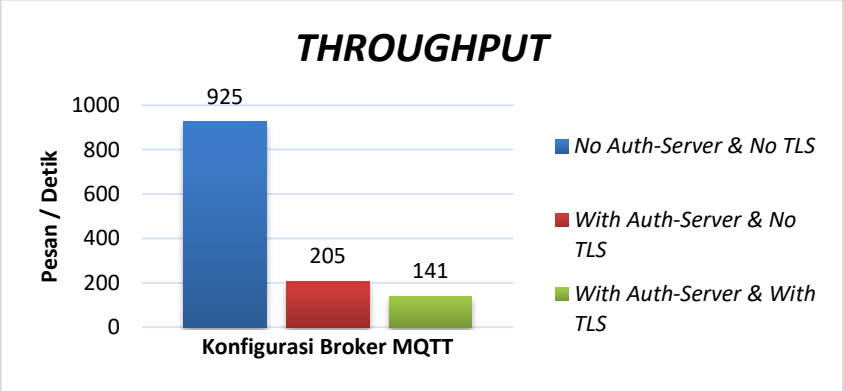
Gambar 6.49 Grafik Hasil Pengujian PPS_001

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 *publish* 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.