

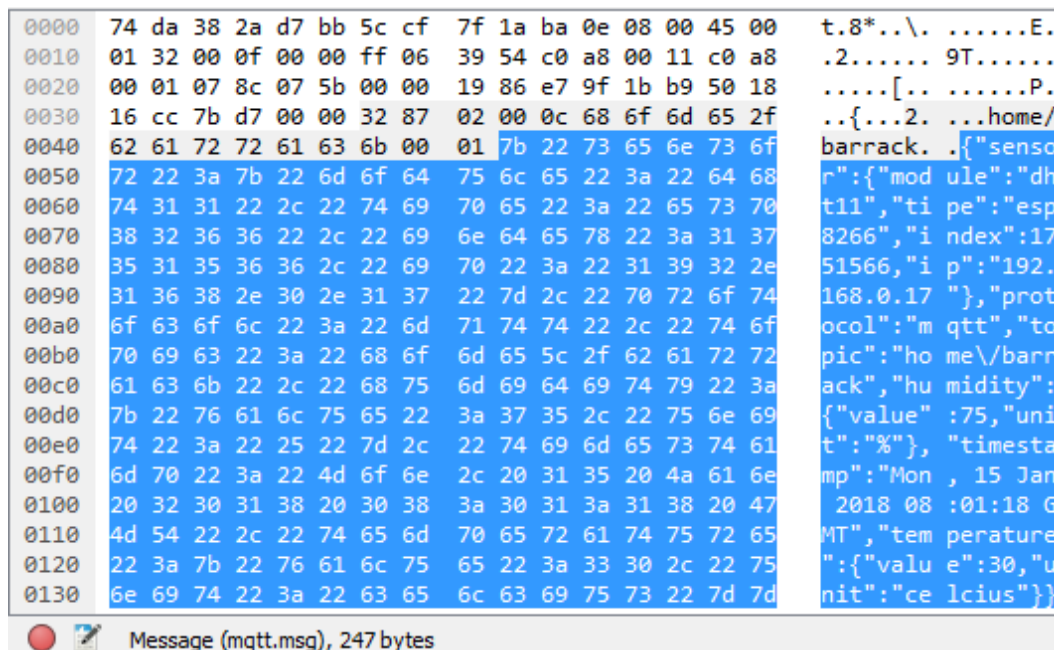
BAB 1 PEMBAHASAN

Berdasarkan pada hasil pengujian dan pengambilan data pada bab sebelumnya, dapat dibahas sebagai berikut:

1.1 Pembahasan Keamanan Data

1.1.1 Skenario 1

Pada pengujian pengiriman data skenario satu, yaitu pengiriman data dengan protokol MQTT tanpa mekanisme keamanan. Menggunakan salah satu teknik MITM yaitu *sniffing*. Dengan menggunakan *program* tcpdump untuk meng-*capture* data dari koneksi wlan0. Data hasil *capture* tcpdump kemudian diolah pada wireshark dan di-*filter* berdasar pada protokol MQTT kemudian didapatkan data *publish message* dan *publish ACK*. Dari *publish message* bisa dilihat data dari sensor nodemcu sebagai *publisher* pada topik *home/barrack* mengirimkan data seperti berikut:



Gambar 1.1 Hasil sniffing protokol MQTT

Seperti yang terlihat pada gambar 6.1 bahwa data yang dikirim dengan protokol MQTT tanpa mekanisme keamanan mudah untuk dibaca oleh penyerang. Sehingga data bisa disalahgunakan dan merugikan korban.

1.1.2 Skenario 2

Pada pengujian pengiriman data skenario dua, yaitu pengiriman data dengan protokol CoAP tanpa mekanisme keamanan. Menggunakan salah satu teknik MITM yaitu *sniffing*. Dengan menggunakan program tcpdump untuk meng-*capture* data dari koneksi wlan0. Data hasil *capture* tcpdump kemudian diolah pada wireshark dan di-*filter* berdasar pada protokol CoAP kemudian didapatkan data CON dan ACK. Dari CON bisa dilihat data dari sensor nodemcu sebagai *publisher* pada topik *home/kitchen* mengirimkan data seperti berikut:

0000	74 da 38 2a d7 bb 5c cf 7f 1a ba 0e 08 00 45 00	t.8*..\E.
0010	01 37 00 0d 00 00 ff 11 39 46 c0 a8 00 11 c0 a8	.7. 9F.
0020	00 01 24 ba 16 33 01 23 8f 84 44 02 42 7c 6e 6f	..\$. . .3.# ..D.B no
0030	64 65 3b 31 39 32 2e 31 36 38 2e 30 2e 31 81 72	de;192.1 68.0.1.r
0040	04 68 6f 6d 65 07 6b 69 74 63 68 65 6e ff 7b 22	.home.ki tchen.{"
0050	73 65 6e 73 6f 72 22 3a 7b 22 6d 6f 64 75 6c 65	sensor": {"module
0060	22 3a 22 64 68 74 32 32 22 2c 22 74 69 70 65 22	":"dht22 ", "tipe"
0070	3a 22 65 73 70 38 32 36 36 22 2c 22 69 6e 64 65	:"esp826 6", "inde
0080	78 22 3a 31 37 35 31 35 36 36 2c 22 69 70 22 3a	x":17515 66, "ip":
0090	22 31 39 32 2e 31 36 38 2e 30 2e 31 37 22 7d 2c	"192.168 .0.17"},
00a0	22 70 72 6f 74 6f 63 6f 6c 22 3a 22 63 6f 61 70	"protoco l": "coap
00b0	22 2c 22 74 6f 70 69 63 22 3a 22 68 6f 6d 65 5c	", "topic ": "home\
00c0	2f 6b 69 74 63 68 65 6e 22 2c 22 68 75 6d 69 64	/kitchen ", "humid
00d0	69 74 79 22 3a 7b 22 76 61 6c 75 65 22 3a 37 35	ity":{"v alue":75
00e0	2c 22 75 6e 69 74 22 3a 22 25 22 7d 2c 22 74 69	, "unit": "%"}, "ti
00f0	6d 65 73 74 61 6d 70 22 3a 22 4d 6f 6e 2c 20 31	mestamp" : "Mon, 1
0100	35 20 4a 61 6e 20 32 30 31 38 20 30 39 3a 35 38	5 Jan 20 18 09:58
0110	3a 34 32 20 47 4d 54 22 2c 22 74 65 6d 70 65 72	:42 GMT" , "temper
0120	61 74 75 72 65 22 3a 7b 22 76 61 6c 75 65 22 3a	ature":{" value":
0130	33 30 2c 22 75 6e 69 74 22 3a 22 63 65 6c 63 69	30, "unit ": "celci
0140	75 73 22 7d 7d	us"}}}

Gambar 1.2 Hasil Sniffing protokol CoAP

Seperti yang terlihat pada gambar 6.2 bahwa data yang dikirim dengan protokol CoAP tanpa mekanisme keamanan mudah untuk dibaca oleh penyerang. Sehingga data bisa disalahgunakan dan merugikan korban.

1.1.3 Skenario 3

Pada pengujian pengiriman data skenario tiga, yaitu pengiriman data dengan protokol MQTT dengan *crypto* AES-CBC 128 bits. Menggunakan salah satu teknik MITM yaitu *sniffing*. Dengan menggunakan *program* tcpdump untuk meng-*capture* data dari koneksi wlan0. Data hasil *capture* tcpdump kemudian diolah pada wireshark dan di-*filter* berdasar pada protokol MQTT kemudian didapatkan data *publish message* dan *publish ACK*. Dari *publish message* bisa dilihat data dari sensor nodemcu sebagai *publisher* pada topik *home/barrack* mengirimkan data seperti berikut:

```

0000 74 da 38 2a d7 bb 5c cf 7f 1a ba 0e 08 00 45 00 t.8*..\ . . . . .E.
0010 01 3b 00 0d 00 00 ff 06 39 4d c0 a8 00 11 c0 a8 .;..... 9M.....
0020 00 01 b9 c4 07 5b 00 00 19 86 3c d0 13 3f 50 18 .....[. .<..?P.
0030 16 cc bf ac 00 00 32 90 02 00 0c 68 6f 6d 65 2f .....2. ...home/
0040 62 61 72 72 61 63 6b 00 01 79 73 a1 f6 f2 c5 61 barrack. .ys....a
0050 eb d2 0d 91 f7 3b 89 fe 14 22 3a fd d7 fe bc ff .....;.. " :.....
0060 9f 5f f0 bb c2 99 b1 3e 5d 88 0e e6 70 e9 89 d5 .._.....> ]...p...
0070 5f ff ce d0 0d c3 12 61 0d 02 b7 72 d7 ce af 1f .....a ...r....
0080 89 f0 c6 9a 2d 89 c7 38 a9 d5 66 7a 58 14 45 91 .....-..8 ..fzX.E.
0090 9c 81 78 2a 92 4e 95 33 e7 b2 3d d8 c7 6b be 49 ..x*.N.3 ..=.k.I
00a0 27 e4 66 9c 9f f3 73 75 d4 b5 eb 95 bd 61 bf c2 '.f...su .....a..
00b0 c2 c2 d4 64 38 ab 1c b0 bf 2b 47 8e bd 03 64 5c ...d8... +G...d\
00c0 85 c9 30 cd 0c fd 6e 2d 78 9f 62 72 21 0f 85 e4 ..0...n- x.br!...
00d0 9c 83 ca ca a8 83 97 a2 fc 92 42 a6 72 94 11 86 ..... ..B.r...
00e0 c3 a4 e5 48 33 5a 57 8d 1e a6 d6 10 1c b6 4c 47 ...H3ZW. ....LG
00f0 e0 b7 71 1e 4c 8d 9d b3 3d 16 42 a7 3b 28 73 38 ..q.L... =.B.;(s8
0100 31 f8 0d ef b8 7c ad bd 67 a7 ce 61 f2 05 da f7 1....|.. g..a....
0110 9f 1b 0e 97 30 95 ed 51 bc 5f 67 53 b5 6d bb 68 ....0..Q _gS.m.h
0120 6e fa 55 67 74 e9 0f fb 1a 94 a9 44 9e c4 a6 1f n.Ugt... ..D....
0130 ec 53 5e ca fa f6 57 0a e5 70 47 03 7c c7 53 2c .S^...W. .pG.|.S,
0140 b8 c7 ab 0c df 95 65 7f 59 .....e. Y

```

Message (mqtt.msg), 256 bytes

Gambar 1.3 Hasil sniffing MQTT crypto

Seperti yang terlihat pada gambar 6.3 bahwa data yang dikirim dengan protokol MQTT dengan *crypto* AES-CBC 128 bits sudah dienkripsi dan sulit dibaca oleh penyerang. Sehingga data telah aman dari segi *confidentiality*.

1.1.4 Skenario 4

Pada pengujian pengiriman data skenario empat, yaitu pengiriman data dengan protokol CoAP dengan *crypto* AES-CBC 128 bits. Menggunakan salah satu teknik MITM yaitu *sniffing*. Dengan menggunakan program tcpdump untuk meng-*capture* data dari koneksi wlan0. Data hasil *capture* tcpdump kemudian diolah pada wireshark dan di-*filter* berdasar pada protokol CoAP kemudian didapatkan data CON dan ACK. Dari CON bisa dilihat data dari sensor nodemcu sebagai publisher pada topik *home/kitchen* mengirimkan data seperti berikut:

0000	74 da 38 2a d7 bb 5c cf 7f 1a ba 0e 08 00 45 00	t.8*...\E.
0010	01 40 00 0a 00 00 ff 11 39 40 c0 a8 00 11 c0 a8	.@..... 9@.....
0020	00 01 89 10 16 33 01 2c 02 8d 44 02 a3 19 6e 6f3., ..D...no
0030	64 65 3b 31 39 32 2e 31 36 38 2e 30 2e 31 81 72	de;192.1 68.0.1.r
0040	04 68 6f 6d 65 07 6b 69 74 63 68 65 6e ff 79 73	.home.ki tchen.ys
0050	a1 f6 f2 c5 61 eb d2 0d 91 f7 3b 89 fe 14 e1 70	...a... ..;...p
0060	99 5b d1 cd 3d 6f 44 aa 07 4a 34 57 ab 1c ba fa	.[.=oD. .J4W...
0070	a0 f1 23 93 d1 c7 22 29 f5 f8 ff 4b 9c a3 9f 8e	..#..." ..K...
0080	54 3f ce f4 28 a5 83 c7 44 79 31 e8 c8 81 74 9d	T?... (... Dy1...t.
0090	e2 c0 42 ba c3 aa ef e0 9f 90 84 e9 8a 4f 71 fb	..B..... ..Oq.
00a0	ca 6b 71 a8 d9 00 9c 0d 47 ef 93 e1 b3 a0 6a 86	.kq..... G....j.
00b0	d1 71 e3 a5 28 b5 c2 4c c8 6f 68 35 dd 58 42 82	.q..(..L .oh5.XB.
00c0	f1 60 66 d6 9e 13 d3 7c c7 d3 e4 bb 59 85 8b 2a	..f.... Y..*
00d0	e8 ba a1 f8 56 5f aa 32 f8 eb 45 1c ba a8 cc b9	...V_2 ..E....
00e0	88 5b f7 ba c9 8a 93 20 61 51 19 26 bb 6e 21 71	.[..... aQ.&.n!q
00f0	0e 08 da da cc d7 b0 5e c7 9c d0 83 9a a1 5e dd^^.
0100	f0 67 9a 0c 2e 3b 4c a6 0d 89 14 2d 59 1d e7 95	.g...;L. ...-Y...
0110	8a 61 51 f9 f2 35 6f 23 e0 34 03 c4 f7 a0 30 dc	.aQ..5o# .4....0.
0120	2c f6 89 71 77 22 c1 38 ea 0b f7 a8 e0 6a fa 72	..qw".8j.r
0130	f9 ff e4 24 ab 7d 69 d6 0d 5a 8c 65 2d 1f b4 f9	...\$.}i. .Z.e-...
0140	bd 10 b2 f9 48 a7 5e 28 18 86 5c 67 29 26	...H.^(..\g)&

Payload (coap.payload), 256 bytes

Gambar 1.4 Hasil sniffing CoAP crypto

Seperti yang terlihat pada gambar 6.4 bahwa data yang dikirim dengan protokol CoAP dengan *crypto* AES-CBC 128 bits sudah dienkripsi dan sulit dibaca oleh penyerang. Sehingga data telah aman dari segi *confidentiality*.

1.1.5 Skenario 5

Pada pengujian pengiriman data skenario lima, yaitu pengiriman data dengan protokol MQTT dengan TLS. Salah satu teknik MITM adalah *sniffing*. Dengan menggunakan program tcpdump untuk meng-*capture* data dari koneksi wlan0. Data hasil *capture* tcpdump kemudian diolah pada wireshark dan di-*filter* berdasar pada TCP kemudian didapatkan data sebagai berikut:

No.	Time	Source	Destination	Info	Delta	Time since reference or first frame
5	4.720583	192.168.0.1	192.168.0.17	8080 → 19404 [FIN, ...]		4.656542 4.720583000
6	4.775060	192.168.0.17	192.168.0.1	19404 → 8080 [RST, ...]		0.054477 4.775060000
7	14.362065	192.168.0.17	192.168.0.1	17928 → 8080 [SYN, ...]		9.587005 14.362065000
8	14.362357	192.168.0.1	192.168.0.17	8080 → 17928 [SYN, ...]		0.000292 14.362357000
9	14.375100	192.168.0.17	192.168.0.1	17928 → 8080 [PSH, ...]		0.012743 14.375100000
10	14.375289	192.168.0.1	192.168.0.17	8080 → 17928 [ACK, ...]		0.000189 14.375289000
11	14.398305	192.168.0.1	192.168.0.17	8080 → 17928 [FIN, ...]		0.023016 14.398305000
12	14.405204	192.168.0.17	192.168.0.1	17928 → 8080 [FIN, ...]		0.006899 14.405204000
13	14.405481	192.168.0.1	192.168.0.17	8080 → 17928 [ACK, ...]		0.000277 14.405481000
14	14.661897	192.168.0.1	192.168.0.17	8080 → 36447 [FIN, ...]		0.256416 14.661897000
15	14.702038	192.168.0.17	192.168.0.1	36447 → 8080 [RST, ...]		0.040141 14.702038000

Gambar 1.5 Hasil sniffing MQTT TLS

Seperti yang terlihat pada gambar 6.5 bahwa sensor nodemcu dengan IP address 192.168.0.17 mencoba terhubung dengan server middleware dengan IP address 192.168.0.1. Tetapi tidak bisa terhubung sehingga proses pengiriman data tidak bisa terjadi sehingga implementasi TLS pada komunikasi protokol MQTT antara sensor nodemcu dengan middleware tidak berhasil.

1.1.6 Skenario 6

Pada pengujian pengiriman data skenario lima, yaitu pengiriman data dengan protocol CoAP dengan TLS. Salah satu teknik MITM adalah *sniffing*. Dengan menggunakan program tcpdump untuk meng-*capture* data dari koneksi wlan0. Data hasil *capture* tcpdump kemudian diolah pada wireshark dan di-*filter* berdasar pada TCP kemudian didapatkan data sebagai berikut:

No.	Time	Source	Destination	Info	Delta	Time since reference or first frame
1	0.000000	192.168.0.17	192.168.0.1	15457 → 8080 [SYN] Seq=...	0.000000	0.00000000
2	0.000373	192.168.0.1	192.168.0.17	8080 → 15457 [SYN, ACK]...	0.000373	0.00037300
3	0.004256	192.168.0.17	192.168.0.1	15457 → 8080 [PSH, ACK]...	0.003883	0.00425600
4	0.004450	192.168.0.1	192.168.0.17	8080 → 15457 [ACK] Seq=...	0.000194	0.00445000
5	0.007745	192.168.0.17	192.168.0.1	8080 → 15457 [FIN, ACK]...	0.003295	0.00774500
6	0.014274	192.168.0.1	192.168.0.1	15457 → 8080 [FIN, ACK]...	0.006529	0.01427400
7	0.014454	192.168.0.1	192.168.0.17	8080 → 15457 [ACK] Seq=...	0.000180	0.01445400
13	10.000094	192.168.0.17	192.168.0.1	10104 → 8080 [SYN] Seq=...	0.969444	10.00009400
14	10.000414	192.168.0.1	192.168.0.17	8080 → 10104 [SYN, ACK]...	0.000320	10.00041400
15	10.004355	192.168.0.17	192.168.0.1	10104 → 8080 [PSH, ACK]...	0.003941	10.00435500
16	10.004554	192.168.0.1	192.168.0.17	8080 → 10104 [ACK] Seq=...	0.000199	10.00455400

Gambar 1.6 Hasil sniffing CoAP TLS

Seperti yang terlihat pada gambar 6.6 bahwa sensor nodemcu dengan IP address 192.168.0.17 mencoba terhubung dengan server middleware dengan IP address 192.168.0.1. Tetapi tidak bisa terhubung sehingga proses pengiriman data tidak bisa terjadi sehingga implementasi TLS pada komunikasi protokol CoAP antara sensor nodemcu dengan *middleware* tidak berhasil.

1.2 Pembahasan Pengiriman Data

1.2.1 Skenario 1

Pada pengujian skenario satu, yaitu pengiriman data dengan protokol MQTT tanpa menggunakan mekanisme keamanan didapat data hasil pengujian sebagai berikut:

Tabel 1.1 Hasil pengujian skenario 1

MQTT					
Expected	Actual	Success rate	Packet loss rate	Delay	Jitter
20	16	80 %	20 %	0.017583	0.00051
20	16	80 %	20 %	0.018451	0.0000094
20	16	80 %	20 %	0.017796	0.00077
20	16	80 %	20 %	0.01959	0.000052
20	16	80 %	20 %	0.016777	0.000035
Average (detik)				0.018039	0.000269

Pada tabel 6.1 terlihat hasil pengujian pengiriman data dengan protokol MQTT dari sensor nodeMcu ke IoT *middleware* tanpa mekanisme keamanan. Dimana rata-rata data terkirim adalah sebanyak 16 data dengan *success rate* 80 % dan *packet loss* 20% dari 4 data. Nilai rata-rata *delay* dari lima kali pengujian adalah 0.018039 detik dan dengan *jitter* adalah 0.000269 detik.

1.2.2 Skenario 2

Pada pengujian skenario dua, yaitu pengiriman data dengan protokol CoAP tanpa menggunakan mekanisme keamanan didapat data hasil pengujian sebagai berikut:

Tabel 1.2 Hasil pengujian skenario 2

CoAP					
Expected	Actual	Success rate	Packet loss rate	Delay	Jitter
20	19	95 %	5 %	0.023074	0.00042
20	19	95 %	5 %	0.019547	0.00081
20	19	95 %	5 %	0.019323	0.00064
20	19	95 %	5 %	0.016898	0.00065
20	18	90 %	10 %	0.019681	0.0008
Average (detik)				0.019705	0.000664

Pada tabel 6.2 terlihat hasil pengujian pengiriman data dengan protokol CoAP dari sensor nodeMcu ke IoT *middleware* tanpa mekanisme keamanan. Dimana rata-rata data terkirim adalah sebanyak 18,8 data dengan *success rate* 94 % dan *packet loss* 6% dari 1,2 data. Nilai rata-rata *delay* dari lima kali pengujian adalah 0.019705 detik dan dengan *jitter* adalah 0.000664 detik.

1.2.3 Skenario 3

Pada pengujian skenario tiga, yaitu pengiriman data dengan protokol MQTT dengan *crypto* AES-CBC 128 bits didapat data hasil pengujian sebagai berikut:

Tabel 1.3 Hasil pengujian skenario 3

MQTT Crypto					
Expected	Actual	Success rate	Packet loss rate	Delay	Jitter
20	16	80 %	20 %	0.017347	0.00094
20	16	80 %	20 %	0.017922	0.00081
20	12	60 %	40 %	0.019189	0.0000032
20	16	80 %	20 %	0.014171	0.000014
20	16	80 %	20 %	0.01651	0.000188
Average (detik)				0.0170278	0.0003894

Pada tabel 6.3 terlihat hasil pengujian pengiriman data dengan protokol MQTT dari sensor nodeMcu ke IoT *middleware* dengan *crypto* AES-CBC 128 bits. Dimana rata-rata data terkirim adalah sebanyak 15,2 data dengan *success rate* 76 % dan *packet loss* 24% dari 4,8 data. Nilai rata-rata *delay* dari lima kali pengujian adalah 0.0170278 detik dan dengan *jitter* adalah 0.0003894 detik.

1.2.4 Skenario 4

Pada pengujian skenario empat, yaitu pengiriman data dengan protokol MQTT dengan *crypto* AES-CBC 128 bits didapat data hasil pengujian sebagai berikut:

Tabel 1.4 Hasil pengujian skenario 4

CoAP Crypto					
Expected	Actual	Success rate	Packet loss rate	Delay	Jitter
20	19	95 %	5 %	0.026884	0.00061
20	19	95 %	5 %	0.020728	0.00082
20	20	100 %	0 %	0.020763	0.00083
20	19	95 %	5 %	0.023594	0.00071
20	19	95 %	5 %	0.024683	0.00074
Average (detik)				0.02333	0.000742

Pada tabel 6.4 terlihat hasil pengujian pengiriman data dengan protokol CoAP dari sensor nodeMcu ke IoT *middleware* dengan *crypto* AES-CBC 128 bits. Dimana rata-rata data terkirim adalah sebanyak 19,2 data dengan *success rate* 96 % dan *packet loss* 4% dari 0,8 data. Nilai rata-rata *delay* dari lima kali pengujian adalah 0.02333 detik dan dengan *jitter* adalah 0.000742 detik.

1.2.5 Skenario 5

Pada pengujian skenario lima, yaitu pengiriman data dengan protokol MQTT dengan TLS. Data tidak berhasil dikirimkan karena client handshake gagal terjadi seperti pada gambar berikut:

```
TCP port is set: 8080.  
TCP ip is set: 192.168.0.1  
socket_connect is called.  
certificate 29d  
client handshake start.  
ip:192.168.0.17,mask:255.255.255.0,gw:192.168.0.1  
net_create is called.  
TCP server/socket is set.  
net_on is called.  
net_on is called.  
net_start is called.  
TCP port is set: 8080.  
TCP ip is set: 192.168.0.1  
socket_connect is called.  
certificate 29d  
client handshake start.
```

Gambar 1.7 Pengiriman data MQTT TLS pada ESPlorer

Terlihat pada gambar 6.7 sertifikat untuk verifikasi TLS sudah tersedia, namun koneksi antara sensor nodemcu ESP8226 dengan server *middleware* tidak berhasil. Ini dibuktikan dengan client handshake start yang terus berulang. Yang artinya server tidak menerima handshake dari klien atau sensor dan berhenti pada tahap client hello.

1.2.6 Skenario 6

Pada pengujian skenario enam, yaitu pengiriman data dengan protokol CoAP dengan TLS. Data tidak berhasil dikirimkan karena client handshake gagal terjadi seperti pada gambar berikut:

```
Connected to CoAP
URL: coap://192.168.0.1:5683/r/home/kitchen
Client ID: 1751566
pm open,type:2 0
client handshake start.
espconn_mbedtls.c 652, type[certificate],length[669]
please start sntp first !
client handshake failed!
Reason:[-0x7880]
```

Gambar 1.8 Pengiriman data CoAP TLS pada ESPlorer

Terlihat pada gambar 6.8 sertifikat untuk verifikasi TLS sudah tersedia, namun koneksi antara sensor nodemcu ESP8226 dengan server middleware tidak berhasil. Ini dibuktikan dengan client handshake failed yang terus berulang. Yang artinya server tidak menerima handshake dari klien atau sensor dan berhenti pada tahap client hello. Dan alasan dari *client handshake failed* adalah sebagai berikut:

```
... 77 #define MBEDTLS_ERR_SSL_PEER_CLOSE_NOTIFY
-0x7880 /**< The peer notified us that the connection is going to be closed. */
```

Gambar 1.9 mbedtls error code

Pada gambar 6.9 dijelaskan bahwa peer SSL memberi pesan bahwa koneksi akan segera ditutup.

1.3 Perbandingan Hasil Pengujian

Tabel 1.5 Perbandingan hasil pengujian

Average			
Mekanisme	Packet loss	Delay	Jitter
MQTT	20 %	0.018039	0.000269
CoAP	6 %	0.019705	0.000664
MQTT-Crypto	24 %	0.0170278	0.0003894
CoAP-Crypto	4 %	0.02333	0.000742

Pada tabel 6.5 terlihat rata-rata packet loss antara tiap protokol sebelum dan sesudah ditambahkan crypto tidak berubah drastis. Pada protokol MQTT adalah 20% sebelum dan 24%

setelah ditambahkan crypto. Sedangkan pada protokol CoAP adalah 6% sebelum dan 4%. Packet loss bisa disebabkan oleh beberapa faktor, yaitu:

1. Sensor nodemcu ESP8266 terkadang berhenti berfungsi dan bisa mengirimkan data kembali setelah interval waktu sekitar 1-2 menit
2. Paket tidak bisa terkirim karena kondisi jaringan.

Nilai delay pada setiap protokol juga hampir tidak berubah. Pada protokol MQTT adalah 0.018039 detik dan 0.0170278 detik setelah ditambahkan crypto. Sedangkan pada protokol CoAP adalah 0.019705 detik sebelum dan 0.02333 detik.

Nilai jitter juga menunjukkan hasil yang serupa. Pada protokol MQTT adalah 0.000269 detik sebelum dan 0.0003894 detik setelah ditambahkan crypto. Dan protokol CoAP adalah 0.000664 detik sebelum dan 0.000742 detik sesudah.

Hal ini terjadi karena data payload yang berisi data temperature dan humidity dienkripsi terlebih pada payload baru kemudian dikirim ke middleware.