

SISTEM TERDISTRIBUSI LINGKUNGAN PRAKTIKUM MANDIRI EKSPLORASI PROTOKOL MQTT, TCP, REST, ZMQ

Nama : Yayan Rachmadian R.
NIM : 256150100111007

1. Protokol MQTT

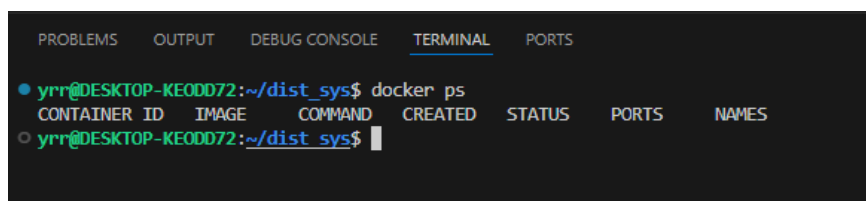
MQTT (Message Queuing Telemetry Transport) merupakan protocol Komunikasi yang yang menggunakan model Broker, Publisher, dan Subscriber. Protokol ini dirancang untuk

- Perangkat dengan resource terbatas (IoT, Embedded System)
- Jaringan tidak stabil/bandwidth kecil (Jaringan Seluler, komunikasi Satelit)

Arsitektur MQTT meliputi Broker, Publisher, dan Subscriber. Model tersebut merupakan:

- Broker
Komponen utama untuk mengatur lalu lintas pesan, contoh: Eclipse Mosquito
- Publisher
Pihak yang mengirim pesan ke sebuah topik, contoh: Sensor suhu mengirim data ke topik
- Subscriber
Pihak yang menerima pesan dari topik tertentu, contoh: aplikasi mobile subscribe akan mendapat update setiap ada data baru

Eksplorasi Protokol MQTT



```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
yrr@DESKTOP-KE00D72:~/dist_sys$ docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
yrr@DESKTOP-KE00D72:~/dist_sys$
```

“docker ps” untuk melihat list docker yang sedang aktif

- Publisher mengirim pesan, broker menerima
- Broker melakukan proses checking siapa subscriber dari topic
- Broker mendistribusikan pesan ke semua subscriber

Proses log untuk mencatat semua proses komunikasi pada protocol MQTT sebagai berikut:

```
yrr@DESKTOP-KE0DD72:~/dist_sys$ ip a
```

Pada terminal ke 3, cek ip untuk mencari kode br yang akan digunakan pada proses mencatat log. "ip a".

```
inet6 fe80::d4a2:5dff:fe70:8ee/64 scope link  
    valid_lft forever preferred_lft forever  
20: veth24c645b@if2: <BROADCAST,MULTICAST,UP,LOWER_UP>  
mtu 1500 qdisc noqueue master br-d22e66d8d792 state UP  
group default
```

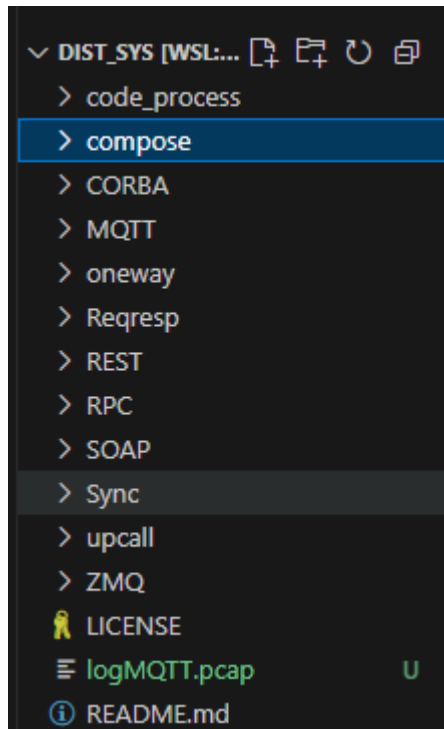
Ambil kode br-d22e66d8d792

```
yrr@DESKTOP-KE0DD72:~/dist_sys$ sudo tcp  
dump -nvi br-d22e66d8d792 -w logMQTT.pca  
p
```

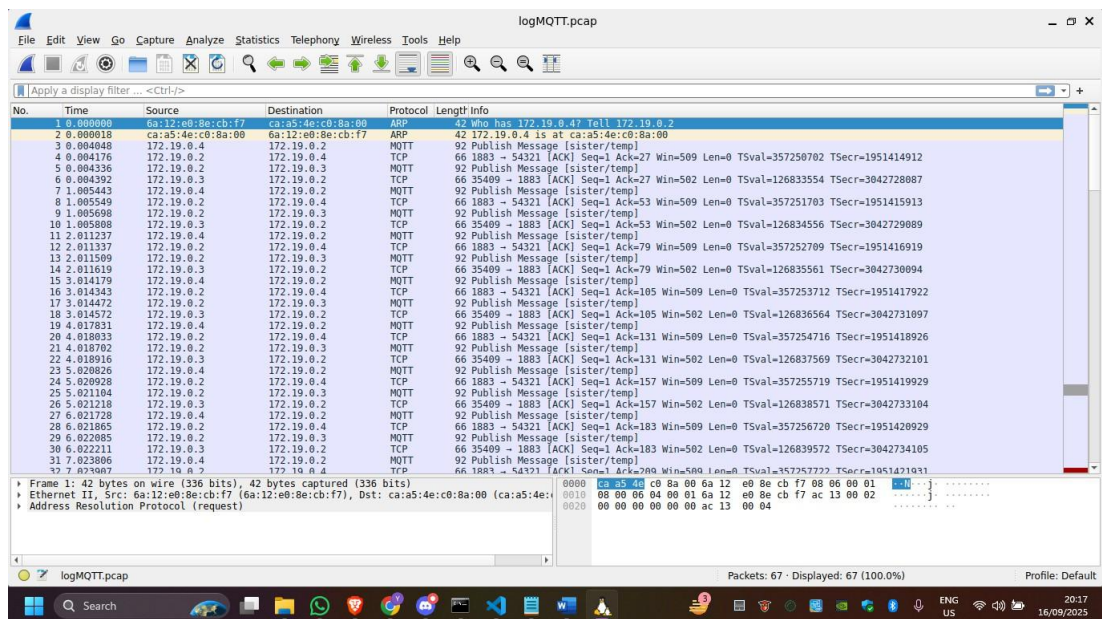
Melakukan pencatatan log dengan perintah "sudo tcpdump -nvi [kode br] -w [namafile log].pcap"

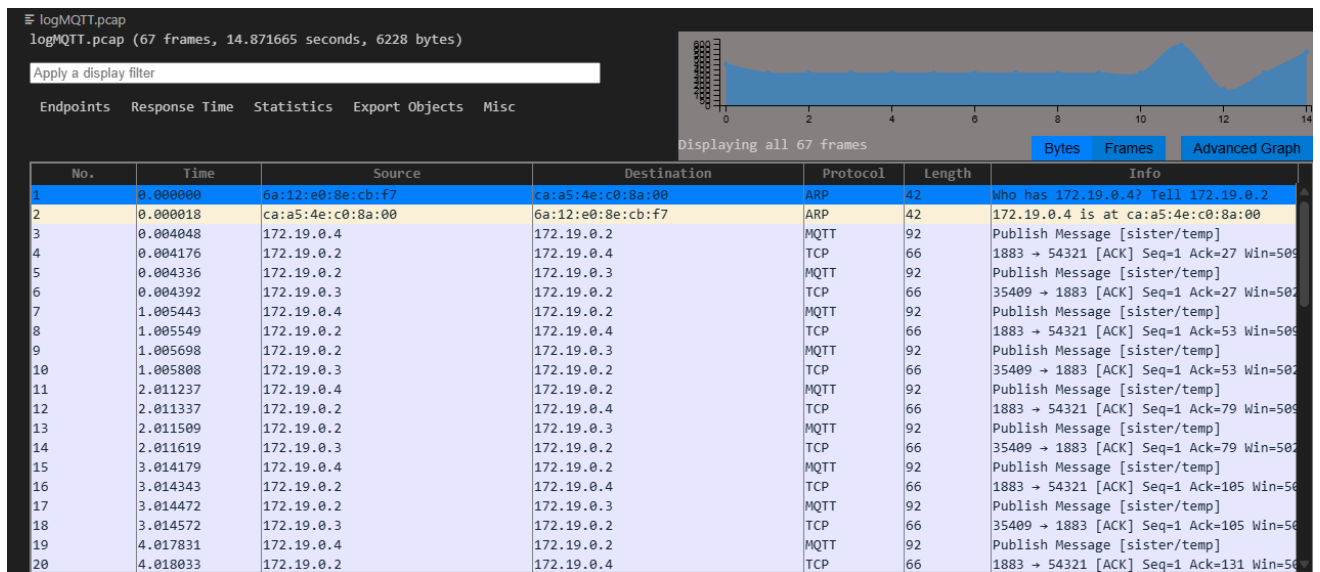
```
yrr@DESKTOP-KE0DD72:~/dist_sys$ sudo tcp  
dump -nvi br-d22e66d8d792 -w logMQTT.pca  
p  
tcpdump: listening on br-d22e66d8d792, l  
ink-type EN10MB (Ethernet), snapshot len  
gth 262144 bytes  
got 67
```

Proses pencatatan log berjalan



Untuk membuka log, Double click "logMQTT.pcap"





a. ARP Request/Reply (Frame 1-2)

- Paket pertama adalah ARP: Who has 172.19.0.4? Tell 172.19.0.2
- Tujuan: mencari MAC address dari host 172.19.0.4.
- Balasan: 172.19.0.4 is at ca:a5:4e:c0:8a:00.
- Menandakan address resolution antar node di jaringan Docker.

b. MQTT Publish

- Terlihat banyak paket Publish Message [sister/temp].
- Source: 172.19.0.2 (Publisher)
- Destination: 172.19.0.4 (Broker atau Subscriber melalui Broker)
- Payload: data sensor yang dipublish ke topic sister/temp.
- Publisher mengirim data ke broker.

c. TCP ACK

- Setelah setiap PUBLISH, ada paket TCP ACK dari broker ke publisher.
- Contoh: 1883 -> 54321 [ACK] Seq=1 Ack=27 Win=502.
- Ini memastikan pesan diterima oleh broker, meskipun MQTT QoS = 0 (at most once).
- Meskipun QoS=0, tetap ada TCP ACK karena MQTT berjalan di atas TCP.

d. Pola Pengiriman

- Publisher (172.19.0.2) mengirim berulang-ulang ke topic sister/temp.
- Subscriber (172.19.0.3) terlihat ikut serta dalam komunikasi (menerima dari broker).
- Pola ini sesuai model Pub/Sub → 1 Publisher, 1 Broker, 1 Subscriber.

2. Protokol TCP (Req/Resp)

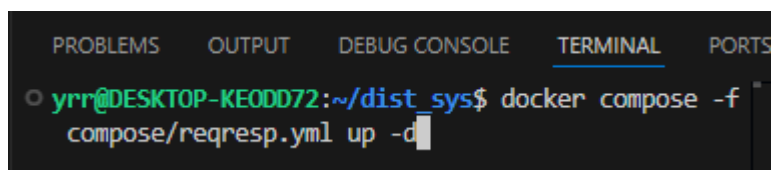
TCP (Transmission Control Protocol) adalah protokol komunikasi pada lapisan transport (OSI Layer 4) yang bersifat connection-oriented dan reliable. Adapun ciri utama dari TCP adalah

- Three-way handshake untuk membuat koneksi (SYN → SYN-ACK → ACK)
- Menjamin urutan paket
- Menjamin tidak ada kehilangan data (dengan ACK & retransmission)
- Mendukung komunikasi full-duplex (dua arah bersamaan)

TCP digunakan dalam pola request/response sederhana

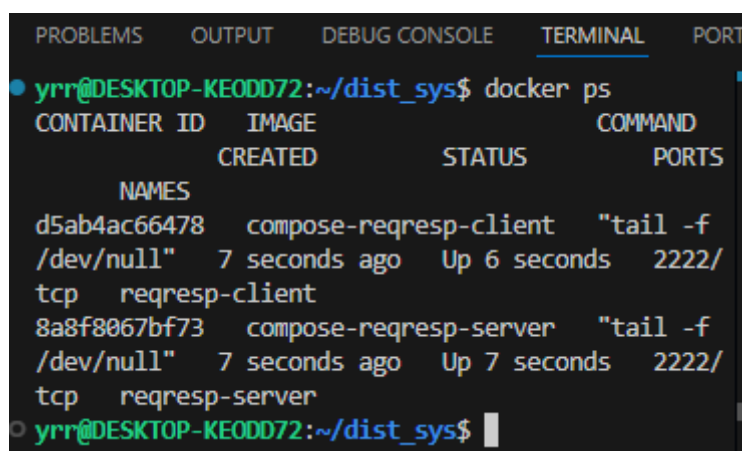
- Server mendengarkan koneksi pada port tertentu (contoh: 2222).
- Client membuat koneksi ke server → terjadi 3-way handshake.
- Client mengirim Request (misalnya pesan teks "halo").
- Server menerima pesan, lalu mengirim Response (misalnya "Echo: halo").
- Proses ini bisa diulang berkali-kali selama koneksi aktif.
- Jika selesai, salah satu pihak melakukan FIN/ACK untuk menutup koneksi.

Eksplorasi Protokol TCP



```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
yrr@DESKTOP-KE0DD72:~/dist_sys$ docker compose -f
compose/reqresp.yml up -d
```

Pada terminal 1, membaca file compose, membuat network internal, menjalankan container, -d berjalan pada background dengan perintah, "docker compose -f compose/reqresp.yml up -d"



```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
yrr@DESKTOP-KE0DD72:~/dist_sys$ docker ps
CONTAINER ID   IMAGE                                COMMAND                  CREATED        STATUS        PORTS
NAMES
d5ab4ac66478   compose-reqresp-client              "tail -f /dev/null"       7 seconds ago Up 6 seconds  2222/tcp
reqresp-client
8a8f8067bf73   compose-reqresp-server              "tail -f /dev/null"       7 seconds ago Up 7 seconds  2222/tcp
reqresp-server
yrr@DESKTOP-KE0DD72:~/dist_sys$
```

List docker ps, tcp berjalan di port 2222

```

yrr@DESKTOP-KE0DD72:~/dist_sys$ docker compose -f compose/reqresp.yml exec reqresp-server python server.py
yrr@DESKTOP-KE0DD72:~/dist_sys$ docker compose -f compose/reqresp.yml exec reqresp-client python client.py

```

Pada terminal 1 bertindak sebagai server, dengan mengetikkan perintah, " docker compose -f compose/reqresp.yml exec reqresp-server python server.py"

Pada terminal 2 bertindak sebagai client, dengan mengetikkan perintah, " docker compose -f compose/reqresp.yml exec reqresp-client python client.py"

```

yrr@DESKTOP-KE0DD72:~/dist_sys$ docker compose -f compose/reqresp.yml exec reqresp-server python server.py
WARN[0000] /home/yrr/dist_sys/compose/reqresp.yml: the attribute 'version' is obsolete, it will be ignored, please remove it to avoid potential confusion
Server listening on 0.0.0.0:2222
Connection from: ('172.19.0.3', 55726)

yrr@DESKTOP-KE0DD72:~/dist_sys$ docker compose -f compose/reqresp.yml exec reqresp-client python client.py
WARN[0000] /home/yrr/dist_sys/compose/reqresp.yml: the attribute 'version' is obsolete, it will be ignored, please remove it to avoid potential confusion
Enter message:

inet6 fe80::8c3b:12ff:fe48:1e1b/64 scope link
    valid_lft forever preferred_lft forever
25: veth9140bd2@if2: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue master br-542d7725c408 state UP group default
    link/ether a6:4b:ce:f5:d5:7c brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet6 fe80::a44b:ceff:fef5:d57c/64 scope link
        valid_lft forever preferred_lft forever
26: vetha47a4c1@if2: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue master br-542d7725c408 state UP group default
    link/ether 06:8f:e6:45:9e:30 brd ff:ff:ff:ff:ff:ff link-netnsid 1
    inet6 fe80::48f:e6ff:fe45:9e30/64 scope link
        valid_lft forever preferred_lft forever
yrr@DESKTOP-KE0DD72:~/dist_sys$

```

Hasil ip addr

24: br-524d7725c408: <...> mtu 1500 qdisc noqueue state UP group default link/ether 8e:3b:01:42:81:0d brd ff:ff:ff:ff:ff:ff

- inet 172.19.0.1/16 brd 172.19.255.255 scope global br-524d7725c408
Interface br-524d7725c408 punya IP **172.19.0.1/16**
- Inilah gateway bridge untuk container 172.19.0.2 (server) dan 172.19.0.3 (client).
- Kode br yang harus dipantau adalah **br-524d7725c408**

```

yrr@DESKTOP-KE0DD72:~/dist_sys$ sudo tcpdump -nvi br-542d7725c408 -w logTCP.pcap
tcpdump: listening on br-542d7725c408, link-type EN10MB (Ethernet), snapshot length 262144 bytes
Got 0

```

Untuk melakukan pencatatan log mengetikkan perintah, "sudo tcpdump -nvi [kode br] -w [nama file].pcap".

```

compose -f compose/reqresp.yml exec reqresp-client python client.py
Received from server: Echo: halo
Enter another message: ingin mencoba untuk mengirim pesan
<class 'bytes'>
Received from server: Echo: ingin mencoba untuk mengirim pesan
Enter another message: apakah berhasil
<class 'bytes'>
Received from server: Echo: apakah berhasil
Enter another message:

```

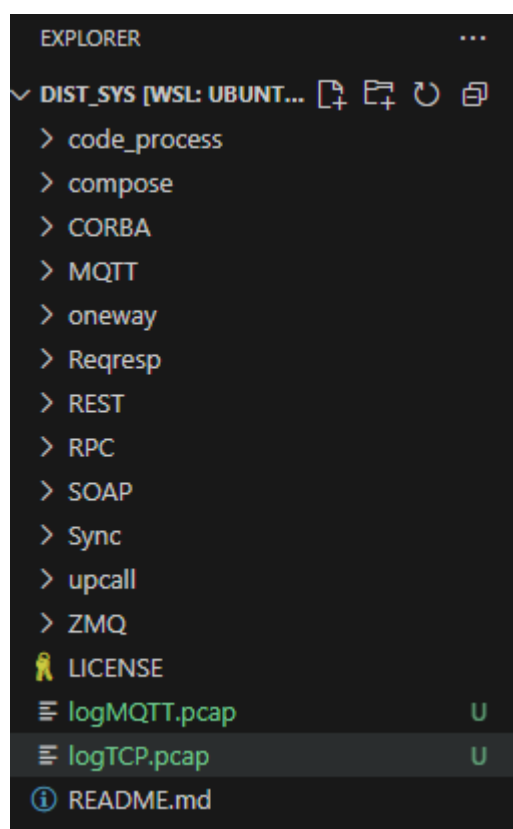
Pada terminal 2 yang bertindak sebagai client berkirir pesan

```
compose -f compose/reqresp.yml exec reqresp-server python server.py
reqresp.yml: the attribute `version` is obsolete, it will be ignored, please remove it to avoid potential confusion
Server listening on 0.0.0.0:2222
Connection from: ('172.19.0.3', 55726)
Received from client: halo
Received from client: ingin mencoba untuk mengirim pesan
Received from client: apakah berhasil
```

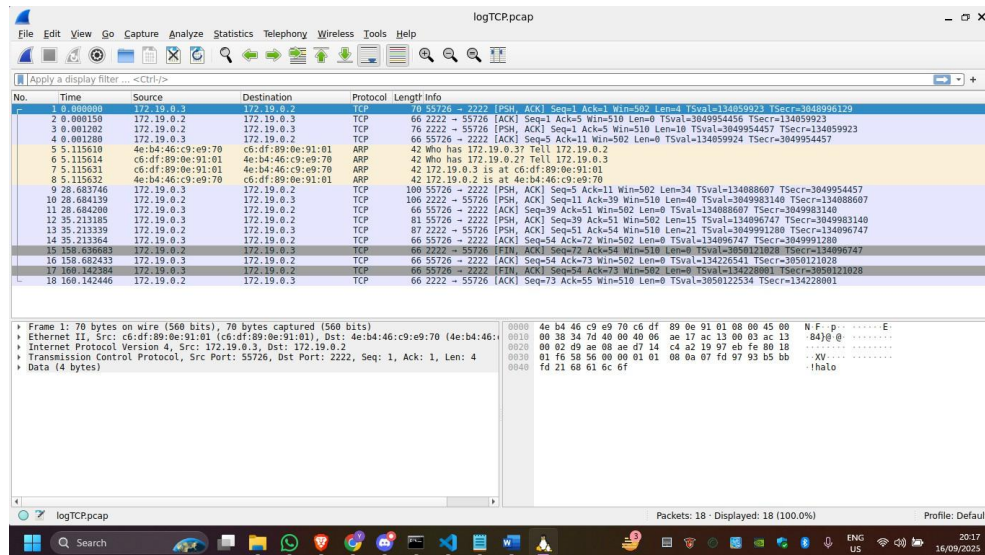
Pada terminal 1 yang bertindak sebagai server menerima pesan yang telah dikirim client

```
yrr@DESKTOP-KE0DD72:~/dist_sys$ sudo tcpdump -nvi br-542d7725c408 -w logTCP.pcap
tcpdump: listening on br-542d7725c408, link-type EN10MB (Ethernet), snapshot length 262144 bytes
Got 14
```

Pada terminal 3 melakukan capture packet



File capture disimpan dalam logTCP.pcap



logTCP.pcap (18 frames, 160.142446 seconds, 1528 bytes)

Apply a display filter

Endpoints Response Time Statistics Export Objects Misc

Displaying all 18 frames

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	172.19.0.3	172.19.0.2	TCP	70	55726 → 2222 [PSH, ACK] Seq=1 Ack=1 Win=502 Len=4 TSval=134059923 TSecr=3048996129
2	0.000150	172.19.0.2	172.19.0.3	TCP	66	2222 → 55726 [ACK] Seq=1 Ack=5 Win=510 Len=0 TSval=3049954456 TSecr=134059923
3	0.001202	172.19.0.2	172.19.0.3	TCP	76	2222 → 55726 [PSH, ACK] Seq=1 Ack=5 Win=510 Len=10 TSval=3049954457 TSecr=134059923
4	0.001280	172.19.0.3	172.19.0.2	TCP	66	55726 → 2222 [ACK] Seq=5 Ack=11 Win=502 Len=0 TSval=134059924 TSecr=3049954457
5	5.115610	4e:b4:46:c9:e9:70	c6:df:89:0e:91:01	ARP	42	Who has 172.19.0.3? Tell 172.19.0.2
6	5.115614	c6:df:89:0e:91:01	4e:b4:46:c9:e9:70	ARP	42	Who has 172.19.0.2? Tell 172.19.0.3
7	5.115631	c6:df:89:0e:91:01	4e:b4:46:c9:e9:70	ARP	42	172.19.0.3 is at c6:df:89:0e:91:01
8	5.115632	4e:b4:46:c9:e9:70	c6:df:89:0e:91:01	ARP	42	172.19.0.2 is at 4e:b4:46:c9:e9:70
9	28.683746	172.19.0.3	172.19.0.2	TCP	100	55726 → 2222 [PSH, ACK] Seq=5 Ack=11 Win=502 Len=34 TSval=134088667 TSecr=3049954457
10	28.684139	172.19.0.2	172.19.0.3	TCP	106	2222 → 55726 [PSH, ACK] Seq=11 Ack=39 Win=510 Len=40 TSval=3049983140 TSecr=134088667
11	28.684200	172.19.0.3	172.19.0.2	TCP	66	55726 → 2222 [ACK] Seq=39 Ack=51 Win=502 Len=0 TSval=134088667 TSecr=3049983140
12	35.213185	172.19.0.3	172.19.0.2	TCP	81	55726 → 2222 [PSH, ACK] Seq=39 Ack=51 Win=502 Len=15 TSval=134096747 TSecr=3049983140
13	35.213339	172.19.0.2	172.19.0.3	TCP	87	2222 → 55726 [PSH, ACK] Seq=51 Ack=54 Win=510 Len=21 TSval=3049991280 TSecr=134096747
14	35.213364	172.19.0.3	172.19.0.2	TCP	66	55726 → 2222 [ACK] Seq=54 Ack=72 Win=502 Len=0 TSval=134096747 TSecr=3049991280
15	158.636683	172.19.0.2	172.19.0.3	TCP	66	2222 → 55726 [FIN, ACK] Seq=72 Ack=54 Win=510 Len=0 TSval=3050121028 TSecr=134096747
16	158.682433	172.19.0.3	172.19.0.2	TCP	66	55726 → 2222 [ACK] Seq=54 Ack=73 Win=502 Len=0 TSval=134226541 TSecr=3050121028
17	160.142384	172.19.0.3	172.19.0.2	TCP	66	55726 → 2222 [FIN, ACK] Seq=54 Ack=73 Win=502 Len=0 TSval=134228001 TSecr=3050121028
18	160.142446	172.19.0.2	172.19.0.3	TCP	66	2222 → 55726 [ACK] Seq=73 Ack=55 Win=510 Len=0 TSval=134228001 TSecr=134228001

a. Inisialisasi Koneksi (Frame 1-4)

- Frame 1: Client (172.19.0.3) → Server (172.19.0.2), PSH, ACK ke port 2222.
- Frame 2–4: ACK balasan antara client dan server.
- Menandakan koneksi TCP sudah terbentuk dan siap bertukar data.

b. ARP Resolution (Frame 5-8)

- Ada beberapa ARP request/reply (Who has 172.19.0.3? Tell 172.19.0.2).
- Digunakan untuk mencari alamat MAC node lain di dalam jaringan Docker.

c. Data Exchange (Frame 9-14)

- Frame 9: Client kirim data (PSH, ACK) → ini adalah Request.
- Frame 10–12: Balasan dari server (PSH, ACK) → ini adalah Response.
- Frame 13–14: ACK tambahan untuk konfirmasi.
- Inti pola Request/Response (Req/Resp) di TCP.

d. Terminasi Koneksi (Frame 15-18)

- Frame 15: Client kirim FIN, ACK → ingin menutup koneksi.
- Frame 16–18: Server balas ACK lalu FIN, ACK, kemudian client balas ACK.
- Koneksi TCP ditutup secara normal (4-way handshake termination).

3. Protokol REST

REST (Representational State Transfer) adalah arsitektur komunikasi untuk sistem terdistribusi yang berbasis HTTP. REST bukan sebagai protokol baru, namun menggunakan HTTP sebagai transport dan aturan untuk mengakses resource.

- Semua resource (data/objek) direpresentasikan dengan URL.
- Operasi dilakukan dengan HTTP method standar
 - GET → membaca data
 - POST → menambahkan data baru
 - PUT → memperbarui data
 - DELETE → menghapus data

Arsitektur REST meliputi 2 komponen utama yaitu:

1. Client
 - Mengirim request ke server melalui HTTP
 - Dapat berupa browser, mobile app, script python
2. Server (REST API)
 - Menyediakan resource (data/layanan)
 - Menerima request dan mengembalikan response berupa status dan data

Eksplorasi Protokol REST

```
yrr@DESKTOP-KE0DD72:~/dist_sys$ docker compose -f compose/rest.yml up -d
```

Melakukan pembacaan file compose, membuat network internal, menjalankan container, -d berjalan pada background, "docker compose -f compose/rest.yml up -d"

```
yrr@DESKTOP-KE0DD72:~/dist_sys$ docker compose -f compose/rest.yml exec rest-server python server.py
```

Menjalankan protokol rest yang bertindak sebagai client, "docker compose -f compose/rest.yml exec rest-server python server.py", pada terminal 1.

```
-f compose/rest.yml exec rest-server python server.py
* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on all addresses (0.0.0.0)
* Running on http://127.0.0.1:5151
* Running on http://172.18.0.2:5151
Press CTRL+C to quit
* Restarting with stat
* Debugger is active!
* Debugger PIN: 879-840-764
```

```

yrr@DESKTOP-KE0DD72:~/dist_sys$ docker compose
-f compose/rest.yml exec rest-client python cli
ent.py --op both -a 7 -b 7

```

Menjalankan protokol rest yang bertindak sebagai client, "docker compose -f compose/rest.yml exec rest-client python client.py", pada terminal 1.

```

yrr@DESKTOP-KE0DD72:~/dist_sys$ ip a | grep ine
t

```

Pada terminal 3, melakukan cek alamat ip dan cek kode br dengan perintah, "ip a | grep inet".

```

inet 172.17.0.1/16 brd 172.17.255.255 scope
global docker0
inet6 fe80::90d1:38ff:fe60:cb9d/64 scope li
nk
inet 172.18.0.1/16 brd 172.18.255.255 scope
global br-f8c6b823f5c9
inet6 fe80::4461:97ff:fe63:7c3b/64 scope li
nk
inet6 fe80::28be:6aff:fed3:ca8c/64 scope li
nk
inet6 fe80::6ce0:a4ff:fedf:ada9/64 scope li
nk
yrr@DESKTOP-KE0DD72:~/dist_sys$

```

Dari perintah tersebut didapatkan kode br-f8c6b823f5c9

```

yrr@DESKTOP-KE0DD72:~/dist_sys$ sudo tcpdump -n
vi br-f8c6b823f5c9 -w rest.pcap

```

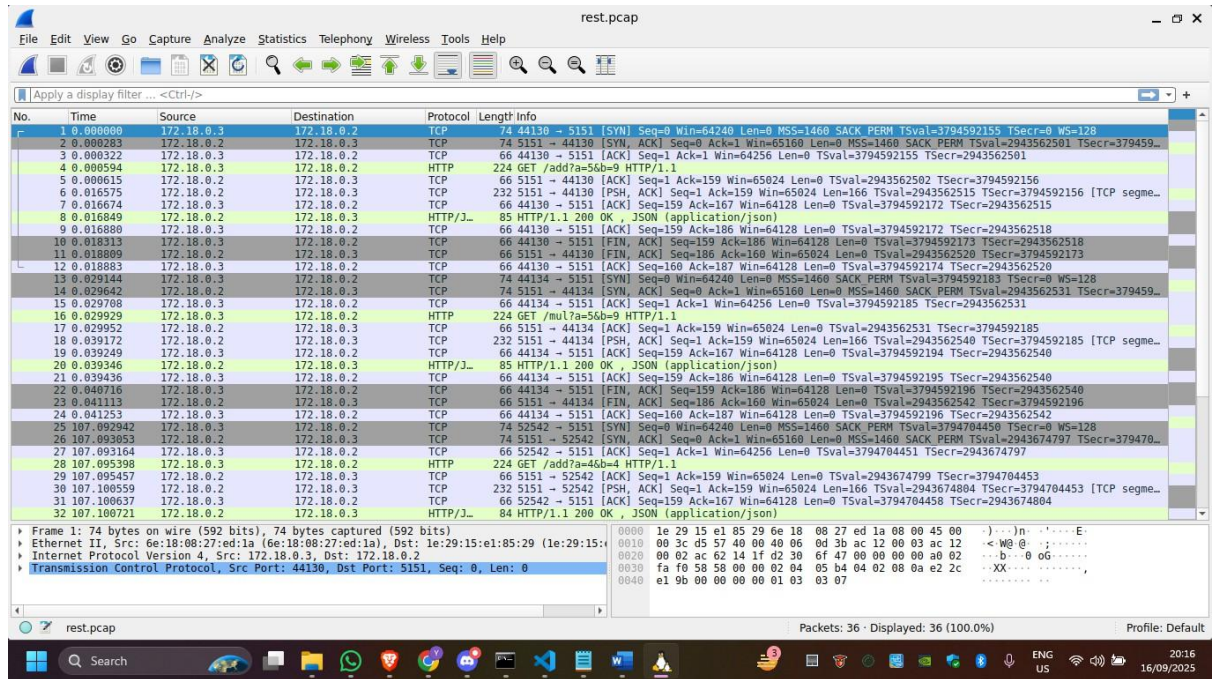
Untuk melakukan pencatatan log mengetikkan perintah, "sudo tcpdump -nvi [kode br] -w [nama file].pcap"

```

yrr@DESKTOP-KE0DD72:~/dist_sys$ sudo tcpdump -n
vi br-f8c6b823f5c9 -w rest.pcap
tcpdump: listening on br-f8c6b823f5c9, link-typ
e EN10MB (Ethernet), snapshot length 262144 byt
es
Got 24

```

Proses pencatatan log pada protokol REST



rest.pcap (52 frames, 112.217706 seconds, 5627 bytes)

Apply a display filter

Endpoints Response Time Statistics Export Objects Misc

Displaying all 52 frames

No.	Time	Source	Destination	Protocol	Length	Info
22	0.040716	172.18.0.3	172.18.0.2	TCP	66	44134 → 5151 [FIN, ACK] Seq=159 Ack=185 Win=0 Len=0
23	0.041113	172.18.0.2	172.18.0.3	TCP	66	5151 → 44134 [FIN, ACK] Seq=186 Ack=160 Win=0 Len=0
24	0.041253	172.18.0.3	172.18.0.2	TCP	66	44134 → 5151 [ACK] Seq=160 Ack=187 Win=0 Len=0
25	107.092942	172.18.0.3	172.18.0.2	TCP	74	52542 → 5151 [SYN] Seq=0 Win=64240 Len=0
26	107.093053	172.18.0.2	172.18.0.3	TCP	74	5151 → 52542 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0
27	107.093164	172.18.0.3	172.18.0.2	TCP	66	52542 → 5151 [ACK] Seq=1 Ack=1 Win=6425 Len=0
28	107.095398	172.18.0.3	172.18.0.2	HTTP	224	GET /add?a=4&b=4 HTTP/1.1
29	107.095457	172.18.0.2	172.18.0.3	TCP	66	5151 → 52542 [ACK] Seq=1 Ack=159 Win=65024 Len=0
30	107.100559	172.18.0.2	172.18.0.3	TCP	232	5151 → 52542 [PSH, ACK] Seq=1 Ack=159 Win=65024 Len=166
31	107.100637	172.18.0.3	172.18.0.2	TCP	66	52542 → 5151 [ACK] Seq=159 Ack=167 Win=64128 Len=0
32	107.100721	172.18.0.2	172.18.0.3	HTTP/JSON	84	HTTP/1.1 200 OK , JSON (application/json)
33	107.100744	172.18.0.3	172.18.0.2	TCP	66	52542 → 5151 [ACK] Seq=159 Ack=185 Win=0 Len=0
34	107.101923	172.18.0.3	172.18.0.2	TCP	66	52542 → 5151 [FIN, ACK] Seq=159 Ack=185 Win=0 Len=0
35	107.102215	172.18.0.2	172.18.0.3	TCP	66	5151 → 52542 [FIN, ACK] Seq=185 Ack=160 Win=0 Len=0
36	107.102255	172.18.0.3	172.18.0.2	TCP	66	52542 → 5151 [ACK] Seq=160 Ack=186 Win=0 Len=0
37	107.106363	172.18.0.3	172.18.0.2	TCP	74	52550 → 5151 [SYN] Seq=0 Win=64240 Len=0
38	107.106462	172.18.0.2	172.18.0.3	TCP	74	5151 → 52550 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0
39	107.106491	172.18.0.3	172.18.0.2	TCP	66	52550 → 5151 [ACK] Seq=1 Ack=1 Win=6425 Len=0
40	107.106622	172.18.0.3	172.18.0.2	HTTP	224	GET /mul?a=4&b=4 HTTP/1.1
41	107.106640	172.18.0.2	172.18.0.3	TCP	66	5151 → 52550 [ACK] Seq=1 Ack=159 Win=65024 Len=0

a. TCP 3-Way Handshake (Frame 1-3)

- Frame 1: Client → Server SYN (port 44318 → 5151).
- Frame 2: Server → Client SYN, ACK.
- Frame 3: Client → Server ACK.
- Koneksi TCP berhasil dibentuk.

b. Request 1 – HTTP GET (Frame 4)

- Client (172.18.0.3) mengirim
GET /add?a=5&b=9 HTTP/1.1
- Client minta resource /add dengan parameter a=5 dan b=9.

- Server (172.18.0.2) merespon (Frame 9):
HTTP/1.1 200 OK
Content-Type: application/json
- Contoh response: {"result":14} (dari 5+9).

c. Request 2 – HTTP GET (Frame 16)

- Client kirim request lain:
GET /mul?a=5&b=9 HTTP/1.1
- Server membalas (Frame 20):
HTTP/1.1 200 OK
Content-Type: application/json
- Contoh response: {"result":45} (dari 5×9).

d. TCP Connection Termination

- Setelah transaksi selesai, ada paket FIN, ACK dari client → server.
- Server balas FIN, ACK.
- Koneksi TCP ditutup dengan normal (4-way handshake).

4. Protokol ZMQ

ZeroMQ (ZMQ) merupakan messaging library berperforma tinggi yang mendukung berbagai pola komunikasi pada sistem terdistribusi. ZeroMQ lebih fleksibel, dapat dipakai untuk point-to-point, publish/subscribe, maupun work distribution.

Pola Komunikasi pada ZMQ ada beberapa macam, yaitu:

a. REQ/REP (Request–Reply)

- Mirip RPC atau client-server tradisional.
- Client (REQ) kirim request → Server (REP) balas reply.
- Berguna untuk komunikasi sinkron.

b. PUB/SUB (Publish–Subscribe)

- Publisher broadcast pesan ke satu atau banyak Subscriber.
- Subscriber bisa filter topik tertentu.
- Cocok untuk notifikasi, sensor data, atau real-time feeds.

c. PUSH/PULL (Pipeline / Work Queue)

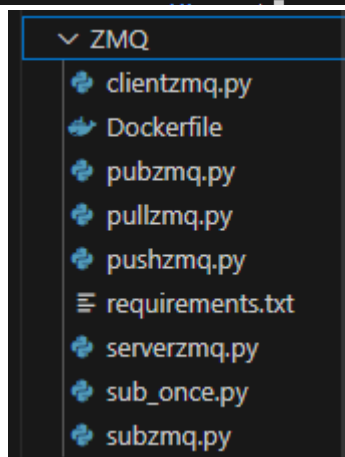
- PUSH mendistribusikan pesan ke beberapa PULL worker.
- Load balancing otomatis: satu pesan hanya diproses satu worker.
- Cocok untuk parallel processing (misalnya job queue).

Eksplorasi Protokol ZMQ

```
○ yrr@DESKTOP-KE0DD72:~/dist_sys$ docker compose
-f compose/zmq.yml up -d
```

Melakukan pembacaan file compose, membuat network internal, menjalankan container, -d berjalan pada background, "docker compose -f compose/zmq.yml up -d"

```
#22 DONE 0.1s
[+] Running 13/13
✓ compose-zmq-pub      Built      0.0s
✓ compose-zmq-sub      Built      0.0s
✓ compose-zmq-push     Built      0.0s
✓ compose-zmq-pull     Built      0.0s
✓ compose-zmq-rep      Built      0.0s
✓ compose-zmq-req      Built      0.0s
✓ Network compose_default Created     0.4s
✓ Container zmq-rep     Started    6.0s
✓ Container zmq-push    Started    5.6s
✓ Container zmq-pub     Started    5.4s
✓ Container zmq-pull    Started    9.9s
✓ Container zmq-sub     Started    9.5s
✓ Container zmq-req     Started    9.6s
```



Dari file .py diatas, pasangan lengkap sehingga membentuk pola komunikasi adalah sebagai berikut:

- clientzmq.py ↔ serverzmq.py (**REQ/REP**)
- pubzmq.py ↔ subzmq.py / sub_once.py (**PUB/SUB**)
- pushzmq.py ↔ pullzmq.py (**PUSH/PULL**)

4.1 Pola Komunikasi ZMQ (Client Server)

```
yrr@DESKTOP-KE0DD72:~/dist_sys$ ip a | grep inet
t
    inet 127.0.0.1/8 scope host lo
    inet 10.255.255.254/32 brd 10.255.255.254 scope global lo
    inet6 ::1/128 scope host
    inet 172.28.34.158/20 brd 172.28.47.255 scope global eth0
    inet6 fe80::215:5dff:fe06:bb27/64 scope link
    inet 172.17.0.1/16 brd 172.17.255.255 scope global docker0
    inet6 fe80::90d1:38ff:fe60:cb9d/64 scope link
    inet 172.18.0.1/16 brd 172.18.255.255 scope global br-7b2205ccd30f
    inet6 fe80::34e4:1dff:fee5:ed57/64 scope link
    inet6 fe80::9cb7:60ff:fe62:c4c3/64 scope link
    inet6 fe80::5c4a:e8ff:fed0:db78/64 scope link
    inet6 fe80::844a:bbff:fe9d:9079/64 scope link
    inet6 fe80::e810:2dff:fe9a:989e/64 scope link
    inet6 fe80::a057:9ff:fe71:c631/64 scope link
    inet6 fe80::ccf8:82ff:fedf:8cf0/64 scope link
yrr@DESKTOP-KE0DD72:~/dist_sys$
```

Untuk melakukan pencatatan log, mengetikkan perintah, "ip a | grep inet", sehingga mendapatkan kode br-7b2205ccd30f

```
yrr@DESKTOP-KE0DD72:~/dist_sys$ docker logs -f zmq-rep
yrr@DESKTOP-KE0DD72:~/dist_sys$ docker compose -f compose/zmq.yml exec zmq-req python clientzmq.py
```

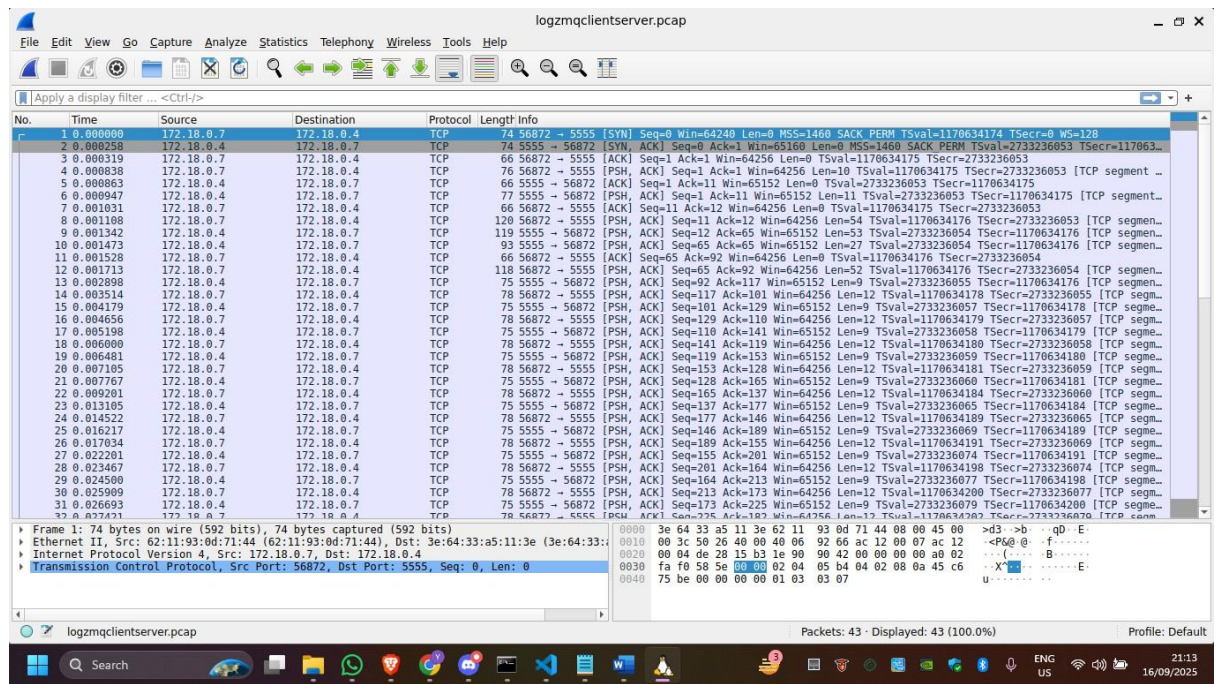
Pada terminal 1 bertindak sebagai server (Rep) dengan perintah, " docker logs -f zmq-rep"
Pada terminal 2 bertindak sebagai client (Req) dengan perintah, "docker compose -f compose/zmq.yml exec zmq-req python clientzmq.py".

```
Received request: b'hello 21 32'
Received request: b'hello 22 33'
Received request: b'hello 23 34'
Received request: b'hello 24 35'
Received request: b'hello 25 36'
Received request: b'hello 26 37'
Received request: b'hello 27 38'
Received request: b'hello 28 39'
Received request: b'hello 29 40'
Received request: b'hello 30 41'
Received request: b'hello 31 42'
Received request: b'hello 32 43'
Received request: b'hello 33 44'
Received request: b'hello 34 45'

-f compose/zmq.yml exec zmq-req python clientzmq.py
Sending request 9 ...
Received reply 9: b'World'
Sending request 10 ...
Received reply 10: b'World'
Sending request 11 ...
Received reply 11: b'World'
Sending request 12 ...
Received reply 12: b'World'
Sending request 13 ...
Received reply 13: b'World'
Sending request 14 ...
Received reply 14: b'World'

yrr@DESKTOP-KE0DD72:~/dist_sys$ sudo tcpdump -n -i br-7b2205ccd30f -w logzmqclientserver.pcap tcpdump: listening on br-7b2205ccd30f, link-type EN10MB (Ethernet), snapshot length 262144 bytes
Got 48
```

Proses komunikasi berjalan dan dicatat pada log.



logmqclientserver.pcap (48 frames, 5.188242 seconds, 4390 bytes)

Apply a display filter

Endpoints Response Time Statistics Export Objects Misc

Displaying all 48 frames

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	172.18.0.7	172.18.0.4	TCP	74	56872 → 5555 [SYN] Seq=0 Win=64240 Len=0
2	0.000258	172.18.0.4	172.18.0.7	TCP	74	5555 → 56872 [SYN, ACK] Seq=0 Ack=1 Win=65152 Len=0
3	0.000319	172.18.0.7	172.18.0.4	TCP	66	56872 → 5555 [ACK] Seq=1 Ack=1 Win=64256 Len=0
4	0.000838	172.18.0.7	172.18.0.4	TCP	76	56872 → 5555 [PSH, ACK] Seq=1 Ack=1 Win=64256 Len=0
5	0.000863	172.18.0.4	172.18.0.7	TCP	66	5555 → 56872 [ACK] Seq=1 Ack=1 Win=65152 Len=0
6	0.000947	172.18.0.4	172.18.0.7	TCP	77	5555 → 56872 [PSH, ACK] Seq=1 Ack=1 Win=65152 Len=0
7	0.001031	172.18.0.7	172.18.0.4	TCP	66	56872 → 5555 [ACK] Seq=11 Ack=12 Win=64256 Len=0
8	0.001108	172.18.0.7	172.18.0.4	TCP	120	56872 → 5555 [PSH, ACK] Seq=11 Ack=12 Win=64256 Len=0
9	0.001342	172.18.0.4	172.18.0.7	TCP	119	5555 → 56872 [PSH, ACK] Seq=12 Ack=65 Win=65152 Len=0
10	0.001473	172.18.0.4	172.18.0.7	TCP	93	5555 → 56872 [PSH, ACK] Seq=65 Ack=92 Win=64256 Len=0
11	0.001528	172.18.0.7	172.18.0.4	TCP	66	56872 → 5555 [ACK] Seq=65 Ack=92 Win=64256 Len=0
12	0.001713	172.18.0.7	172.18.0.4	TCP	118	56872 → 5555 [PSH, ACK] Seq=65 Ack=92 Win=64256 Len=0
13	0.002898	172.18.0.4	172.18.0.7	TCP	75	5555 → 56872 [PSH, ACK] Seq=92 Ack=117 Win=65152 Len=0
14	0.003514	172.18.0.7	172.18.0.4	TCP	78	56872 → 5555 [PSH, ACK] Seq=117 Ack=101 Win=64256 Len=12
15	0.004179	172.18.0.4	172.18.0.7	TCP	75	5555 → 56872 [PSH, ACK] Seq=101 Ack=129 Win=65152 Len=9
16	0.004656	172.18.0.4	172.18.0.7	TCP	78	56872 → 5555 [PSH, ACK] Seq=129 Ack=110 Win=64256 Len=12
17	0.005198	172.18.0.4	172.18.0.7	TCP	75	5555 → 56872 [PSH, ACK] Seq=110 Ack=141 Win=65152 Len=9
18	0.006000	172.18.0.7	172.18.0.4	TCP	78	56872 → 5555 [PSH, ACK] Seq=141 Ack=119 Win=64256 Len=12
19	0.006481	172.18.0.4	172.18.0.7	TCP	75	5555 → 56872 [PSH, ACK] Seq=119 Ack=153 Win=65152 Len=9
20	0.007105	172.18.0.7	172.18.0.4	TCP	78	56872 → 5555 [PSH, ACK] Seq=153 Ack=128 Win=64256 Len=12

a. TCP 3-Way Handshake (Frame 1–3)

- Frame 1: Client → Server SYN (port 56872 → 5555).
- Frame 2: Server → Client SYN, ACK.
- Frame 3: Client → Server ACK.
- Koneksi TCP berhasil dibuat.

b. Data Exchange (Frame 4–40)

- Terlihat banyak paket dengan flag PSH, ACK dari client ke server (56872 → 5555) dan sebaliknya.
- Pola ini konsisten dengan Request/Reply:
 - Client (REQ socket) kirim pesan.
 - Server (REP socket) balas pesan.
- Contoh:
 - Frame 4: Client kirim payload ke server.
 - Frame 5: Server ACK → lalu balas (Frame 6).
 - Frame 7 dst: Pola request → response berulang.

- Menandakan aplikasi berjalan dengan pola sinkron: client menunggu reply sebelum kirim request berikutnya.

c. Tidak ada ARP atau broadcast

- Semua komunikasi langsung point-to-point antara dua host, karena sudah ada koneksi TCP established

d. Tidak terlihat FIN (Termination)

- Dari capture, komunikasi masih berlangsung (belum ada FIN, ACK).
- Artinya client-server tetap terhubung untuk komunikasi lanjutan.

4.2 Pola Komunikasi ZMQ (Push Pull)

The image shows a Wireshark packet capture for the file 'logzmqpushpull.pcap'. The interface displays a list of 46 packets in the packet list pane. The selected packet (No. 1) is an ARP request from 172.18.0.4 to the broadcast address 36:ce:1e:ee:4c:b6. The packet details pane shows the Ethernet II header and the ARP request structure. The packet bytes pane displays the raw data in hexadecimal and ASCII. The status bar at the bottom indicates that 46 packets are displayed (100.0%) and the profile is set to Default.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	ba:4b:9c:43:eb:34	Broadcast	ARP	42	Who has 172.18.0.4? Tell 172.18.0.6
2	0.000167	36:ce:1e:ee:4c:b6	ba:4b:9c:43:eb:34	ARP	42	172.18.0.4 is at 36:ce:1e:ee:4c:b6
3	0.000308	172.18.0.6	172.18.0.4	TCP	74	59726 → 9999 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM TSval=3519251766 TSecr=0 WS=128
4	0.001146	172.18.0.4	172.18.0.6	TCP	74	9999 → 59726 [SYN, ACK] Seq=0 Ack=1 Win=65152 Len=0 MSS=1460 SACK_PERM TSval=1252088781 TSecr=3519251772
5	0.001301	172.18.0.6	172.18.0.4	TCP	66	59726 → 9999 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=3519251771 TSecr=1252088781
6	0.002862	172.18.0.6	172.18.0.4	TCP	76	59726 → 9999 [PSH, ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=3519251772 TSecr=1252088781
7	0.002905	172.18.0.4	172.18.0.6	TCP	66	9999 → 59726 [ACK] Seq=1 Ack=11 Win=65152 Len=0 TSval=1252088783 TSecr=3519251772
8	0.003398	172.18.0.4	172.18.0.6	TCP	77	9999 → 59726 [PSH, ACK] Seq=1 Ack=11 Win=65152 Len=11 TSval=1252088783 TSecr=3519251772
9	0.003508	172.18.0.6	172.18.0.4	TCP	66	59726 → 9999 [ACK] Seq=11 Ack=12 Win=64256 Len=0 TSval=3519251773 TSecr=1252088783
10	0.003678	172.18.0.6	172.18.0.4	TCP	120	59726 → 9999 [PSH, ACK] Seq=11 Ack=12 Win=64256 Len=54 TSval=1252088783 TSecr=1252088783
11	0.004056	172.18.0.4	172.18.0.6	TCP	119	9999 → 59726 [PSH, ACK] Seq=12 Ack=65 Win=65152 Len=53 TSval=1252088784 TSecr=3519251773
12	0.004229	172.18.0.4	172.18.0.6	TCP	94	9999 → 59726 [PSH, ACK] Seq=65 Ack=65 Win=65152 Len=28 TSval=1252088784 TSecr=3519251773
13	0.004283	172.18.0.6	172.18.0.4	TCP	94	59726 → 9999 [PSH, ACK] Seq=65 Ack=93 Win=64256 Len=28 TSval=3519251774 TSecr=1252088784
14	0.004973	172.18.0.4	172.18.0.6	TCP	73	9999 → 59726 [PSH, ACK] Seq=93 Ack=93 Win=65152 Len=7 TSval=1252088785 TSecr=3519251774
15	0.050342	172.18.0.6	172.18.0.4	TCP	66	59726 → 9999 [ACK] Seq=93 Ack=100 Win=64256 Len=0 TSval=3519251820 TSecr=1252088785
16	5.091964	36:ce:1e:ee:4c:b6	ba:4b:9c:43:eb:34	ARP	42	Who has 172.18.0.6? Tell 172.18.0.4
17	5.092004	ba:4b:9c:43:eb:34	36:ce:1e:ee:4c:b6	ARP	42	172.18.0.6 is at ba:4b:9c:43:eb:34
18	12.810041	172.18.0.4	172.18.0.6	TCP	73	9999 → 59726 [PSH, ACK] Seq=100 Ack=93 Win=65152 Len=7 TSval=1252101590 TSecr=3519251820
19	12.810154	172.18.0.6	172.18.0.4	TCP	66	59726 → 9999 [ACK] Seq=93 Ack=107 Win=64256 Len=0 TSval=3519264580 TSecr=1252101590
20	15.812587	172.18.0.4	172.18.0.6	TCP	73	9999 → 59726 [PSH, ACK] Seq=107 Ack=93 Win=65152 Len=7 TSval=1252104592 TSecr=3519264580

The image shows a Wireshark packet capture for the file 'logzmqpushpull.pcap'. The interface displays a list of 46 packets in the packet list pane. The selected packet (No. 1) is an ARP request from 172.18.0.4 to the broadcast address 36:ce:1e:ee:4c:b6. The packet details pane shows the Ethernet II header and the ARP request structure. The packet bytes pane displays the raw data in hexadecimal and ASCII. The status bar at the bottom indicates that 46 packets are displayed (100.0%) and the profile is set to Default.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	ba:4b:9c:43:eb:34	Broadcast	ARP	42	Who has 172.18.0.4? Tell 172.18.0.6
2	0.000167	36:ce:1e:ee:4c:b6	ba:4b:9c:43:eb:34	ARP	42	172.18.0.4 is at 36:ce:1e:ee:4c:b6
3	0.000308	172.18.0.6	172.18.0.4	TCP	74	59726 → 9999 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM TSval=3519251766 TSecr=0 WS=128
4	0.001146	172.18.0.4	172.18.0.6	TCP	74	9999 → 59726 [SYN, ACK] Seq=0 Ack=1 Win=65152 Len=0 MSS=1460 SACK_PERM TSval=1252088781 TSecr=3519251772
5	0.001301	172.18.0.6	172.18.0.4	TCP	66	59726 → 9999 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=3519251771 TSecr=1252088781
6	0.002862	172.18.0.6	172.18.0.4	TCP	76	59726 → 9999 [PSH, ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=3519251772 TSecr=1252088781
7	0.002905	172.18.0.4	172.18.0.6	TCP	66	9999 → 59726 [ACK] Seq=1 Ack=11 Win=65152 Len=0 TSval=1252088783 TSecr=3519251772
8	0.003398	172.18.0.4	172.18.0.6	TCP	77	9999 → 59726 [PSH, ACK] Seq=1 Ack=11 Win=65152 Len=11 TSval=1252088783 TSecr=3519251772
9	0.003508	172.18.0.6	172.18.0.4	TCP	66	59726 → 9999 [ACK] Seq=11 Ack=12 Win=64256 Len=0 TSval=3519251773 TSecr=1252088783
10	0.003678	172.18.0.6	172.18.0.4	TCP	120	59726 → 9999 [PSH, ACK] Seq=11 Ack=12 Win=64256 Len=54 TSval=1252088783 TSecr=1252088783
11	0.004056	172.18.0.4	172.18.0.6	TCP	119	9999 → 59726 [PSH, ACK] Seq=12 Ack=65 Win=65152 Len=53 TSval=1252088784 TSecr=3519251773
12	0.004229	172.18.0.4	172.18.0.6	TCP	94	9999 → 59726 [PSH, ACK] Seq=65 Ack=65 Win=65152 Len=28 TSval=1252088784 TSecr=3519251773
13	0.004283	172.18.0.6	172.18.0.4	TCP	94	59726 → 9999 [PSH, ACK] Seq=65 Ack=93 Win=64256 Len=28 TSval=3519251774 TSecr=1252088784
14	0.004973	172.18.0.4	172.18.0.6	TCP	73	9999 → 59726 [PSH, ACK] Seq=93 Ack=93 Win=65152 Len=7 TSval=1252088785 TSecr=3519251774
15	0.050342	172.18.0.6	172.18.0.4	TCP	66	59726 → 9999 [ACK] Seq=93 Ack=100 Win=64256 Len=0 TSval=3519251820 TSecr=1252088785
16	5.091964	36:ce:1e:ee:4c:b6	ba:4b:9c:43:eb:34	ARP	42	Who has 172.18.0.6? Tell 172.18.0.4
17	5.092004	ba:4b:9c:43:eb:34	36:ce:1e:ee:4c:b6	ARP	42	172.18.0.6 is at ba:4b:9c:43:eb:34
18	12.810041	172.18.0.4	172.18.0.6	TCP	73	9999 → 59726 [PSH, ACK] Seq=100 Ack=93 Win=65152 Len=7 TSval=1252101590 TSecr=3519251820
19	12.810154	172.18.0.6	172.18.0.4	TCP	66	59726 → 9999 [ACK] Seq=93 Ack=107 Win=64256 Len=0 TSval=3519264580 TSecr=1252101590
20	15.812587	172.18.0.4	172.18.0.6	TCP	73	9999 → 59726 [PSH, ACK] Seq=107 Ack=93 Win=65152 Len=7 TSval=1252104592 TSecr=3519264580

a. ARP Resolution (Frame 1–2, 16–17)

- Who has 172.18.0.4? Tell 172.18.0.6 → untuk mencari MAC address node tujuan.
- Balasan: 172.18.0.4 is at 36:ce:1e:....
- bagian awal komunikasi untuk address resolution.

b. TCP Handshake (Frame 3–4)

- Client (172.18.0.6 / Push) → Server (172.18.0.4 / Pull): SYN.
- Server → Client: SYN, ACK.
- Client → Server: ACK.
- Koneksi TCP berhasil dibuat di port 9999.

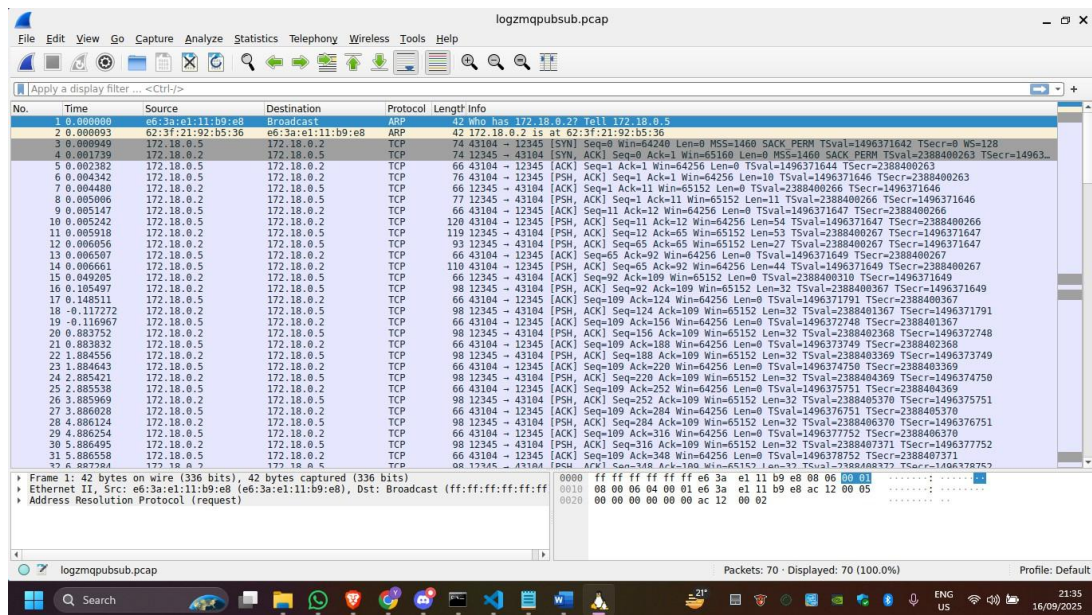
c. Data Transmission (Frame 5–15, 18–46)

- Push node (172.18.0.6) mengirim banyak paket PSH, ACK ke Pull node (172.18.0.4).
- Pull node membalas dengan ACK.
- Tidak ada paket balasan payload dari Pull → karena Push hanya mengirim, Pull hanya menerima (satu arah).
- Contoh pola:
 - Frame 5: 172.18.0.6 → 172.18.0.4 (data dikirim).
 - Frame 6: 172.18.0.4 → 172.18.0.6 (ACK).
 - Frame berikutnya mengulangi pola ini.

d. Koneksi Tetap Terbuka

- Dari frame terakhir, belum ada FIN atau RST. Artinya Push–Pull channel masih aktif untuk terus mendistribusikan data.

4.3 Alur Komunikasi ZMQ (PUB SUB)



logzmqpubsub.pcap (70 frames, 256.122424 seconds, 6722 bytes)

Apply a display filter

Endpoints Response Time Statistics Export Objects Misc

Displaying all 70 frames

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	e6:3a:e1:11:b9:e8	Broadcast	ARP	42	who has 172.18.0.2? Tell 172.18.0.5
2	0.000093	62:3f:21:92:b5:36	e6:3a:e1:11:b9:e8	ARP	42	172.18.0.2 is at 62:3f:21:92:b5:36
3	0.000949	172.18.0.5	172.18.0.2	TCP	74	43104 → 12345 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM TSval=1496371642 TSecr=0 WS=128
4	0.001739	172.18.0.2	172.18.0.5	TCP	74	12345 → 43104 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460 SACK_PERM TSval=2388400263 TSecr=1496371646
5	0.002382	172.18.0.5	172.18.0.2	TCP	66	43104 → 12345 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=1496371644 TSecr=2388400263
6	0.004342	172.18.0.5	172.18.0.2	TCP	76	43104 → 12345 [PSH, ACK] Seq=1 Ack=1 Win=64256 Len=10 TSval=1496371646 TSecr=2388400263
7	0.004480	172.18.0.2	172.18.0.5	TCP	66	12345 → 43104 [ACK] Seq=1 Ack=11 Win=65152 Len=0 TSval=2388400266 TSecr=1496371646
8	0.005006	172.18.0.2	172.18.0.5	TCP	77	12345 → 43104 [PSH, ACK] Seq=1 Ack=11 Win=65152 Len=11 TSval=2388400266 TSecr=1496371646
9	0.005147	172.18.0.5	172.18.0.2	TCP	66	43104 → 12345 [ACK] Seq=11 Ack=12 Win=64256 Len=0 TSval=1496371647 TSecr=2388400266
10	0.005242	172.18.0.5	172.18.0.2	TCP	120	43104 → 12345 [PSH, ACK] Seq=11 Ack=12 Win=64256 Len=54 TSval=1496371647 TSecr=2388400266
11	0.005918	172.18.0.2	172.18.0.5	TCP	119	12345 → 43104 [PSH, ACK] Seq=12 Ack=65 Win=65152 Len=53 TSval=2388400267 TSecr=1496371647
12	0.006056	172.18.0.2	172.18.0.5	TCP	93	12345 → 43104 [PSH, ACK] Seq=12 Ack=65 Win=65152 Len=27 TSval=2388400267 TSecr=1496371647
13	0.006507	172.18.0.5	172.18.0.2	TCP	66	43104 → 12345 [ACK] Seq=65 Ack=92 Win=64256 Len=0 TSval=1496371649 TSecr=2388400267
14	0.006661	172.18.0.5	172.18.0.2	TCP	110	43104 → 12345 [PSH, ACK] Seq=65 Ack=92 Win=64256 Len=44 TSval=1496371649 TSecr=2388400267
15	0.049205	172.18.0.2	172.18.0.5	TCP	66	12345 → 43104 [ACK] Seq=92 Ack=109 Win=65152 Len=0 TSval=2388400310 TSecr=1496371649
16	0.105497	172.18.0.2	172.18.0.5	TCP	98	12345 → 43104 [PSH, ACK] Seq=92 Ack=109 Win=65152 Len=32 TSval=2388400367 TSecr=1496371649
17	0.148511	172.18.0.5	172.18.0.2	TCP	66	43104 → 12345 [ACK] Seq=109 Ack=124 Win=64256 Len=0 TSval=1496371791 TSecr=2388400367
18	-0.117272	172.18.0.2	172.18.0.5	TCP	98	12345 → 43104 [PSH, ACK] Seq=124 Ack=109 Win=65152 Len=32 TSval=2388401367 TSecr=1496371791
19	-0.116967	172.18.0.5	172.18.0.2	TCP	66	43104 → 12345 [ACK] Seq=109 Ack=156 Win=64256 Len=0 TSval=1496372748 TSecr=2388401367
20	0.883752	172.18.0.2	172.18.0.5	TCP	66	12345 → 43104 [PSH, ACK] Seq=156 Ack=109 Win=65152 Len=32 TSval=2388402368 TSecr=1496372748
21	0.883832	172.18.0.5	172.18.0.2	TCP	66	43104 → 12345 [ACK] Seq=109 Ack=188 Win=64256 Len=0 TSval=1496373749 TSecr=2388402368
22	1.884556	172.18.0.2	172.18.0.5	TCP	98	12345 → 43104 [PSH, ACK] Seq=188 Ack=109 Win=65152 Len=32 TSval=2388403369 TSecr=1496373749
23	1.884643	172.18.0.5	172.18.0.2	TCP	66	43104 → 12345 [ACK] Seq=109 Ack=220 Win=64256 Len=0 TSval=1496374750 TSecr=2388403369
24	2.885421	172.18.0.2	172.18.0.5	TCP	98	12345 → 43104 [PSH, ACK] Seq=220 Ack=109 Win=65152 Len=32 TSval=2388404369 TSecr=1496374750
25	2.885538	172.18.0.5	172.18.0.2	TCP	66	43104 → 12345 [ACK] Seq=109 Ack=252 Win=64256 Len=0 TSval=1496375751 TSecr=2388404369
26	3.885969	172.18.0.2	172.18.0.5	TCP	98	12345 → 43104 [PSH, ACK] Seq=252 Ack=109 Win=65152 Len=32 TSval=2388405370 TSecr=1496375751
27	3.886020	172.18.0.2	172.18.0.2	TCP	66	43104 → 12345 [ACK] Seq=109 Ack=284 Win=64256 Len=0 TSval=1496376751 TSecr=2388405370
28	4.886124	172.18.0.2	172.18.0.5	TCP	98	12345 → 43104 [PSH, ACK] Seq=284 Ack=109 Win=65152 Len=32 TSval=2388406370 TSecr=1496376751
29	4.886254	172.18.0.5	172.18.0.2	TCP	66	43104 → 12345 [ACK] Seq=109 Ack=316 Win=64256 Len=0 TSval=1496377752 TSecr=2388406370
30	5.886495	172.18.0.2	172.18.0.5	TCP	98	12345 → 43104 [PSH, ACK] Seq=316 Ack=109 Win=65152 Len=32 TSval=2388407371 TSecr=1496377752
31	5.886558	172.18.0.5	172.18.0.2	TCP	66	43104 → 12345 [ACK] Seq=109 Ack=348 Win=64256 Len=0 TSval=1496378752 TSecr=2388407371
32	6.887284	172.18.0.2	172.18.0.5	TCP	98	12345 → 43104 [PSH, ACK] Seq=348 Ack=109 Win=65152 Len=32 TSval=2388408372 TSecr=1496378752

a. ARP Resolution (Frame 1–2)

- Who has 172.18.0.2? Tell 172.18.0.5 → Subscriber mencari alamat MAC Publisher.
- Balasan: 172.18.0.2 is at 62:3f:21:92:b5:36.
- tahap awal normal untuk menemukan node tujuan.

b. TCP Handshake (Frame 3–4)

- Subscriber (172.18.0.5) → Publisher (172.18.0.2): SYN.
- Publisher → Subscriber: SYN, ACK.
- Subscriber → Publisher: ACK.
- TCP connection terbentuk di port 12345.

c. Data Transmission (Frame 5–70)

- Publisher (172.18.0.2) mengirim banyak paket PSH, ACK ke Subscriber (172.18.0.5).
- Subscriber membalas dengan ACK → konfirmasi penerimaan.
- Pola ini unidirectional (satu arah): hanya Publisher → Subscriber.
- Payload yang dikirim berupa pesan broadcast (konten tidak terlihat langsung di Wireshark, tapi ada di layer aplikasi ZMQ).
- sesuai dengan pola PUB/SUB: Publisher mengirim ke semua Subscriber yang berlangganan topik.

d. Tidak ada Balasan Payload dari Subscriber

- Subscriber hanya memberikan TCP ACK, tidak ada pesan balik.
- Karena PUB/SUB ZeroMQ memang asinkron dan satu arah: Publisher tidak menunggu balasan.

Referensi Kode Program

https://github.com/abazh/dist_sys