

Proyecto 1. Uso de un protocolo existente

Francis Aguilar 22243

Mcps

1. Local

- **Nombre:** Taylor
- **Transporte:** stdio

Herramientas

get_song_lyrics(song_title: str)

- **Parámetro:**
 - song_title: título de la canción de Taylor Swift.
- **Respuesta:**
 - Preview (primeras 4 líneas de la canción).
 - Mensajes de error si no hay resultados.
- **Uso:** Obtener la letra (solo vista previa) para evitar problemas de copyright.

analyze_song(song_title: str)

- **Parámetro:**
 - song_title: título de la canción.
- **Respuesta:**
 - Reporte completo con:
 - estadísticas básicas (palabras, líneas, densidad, repetición),
 - perfil emocional (positivo, negativo, romántico, intensidad),
 - características de la canción (romántica, melancólica, alegre, repetitiva, complejidad),
 - palabras más frecuentes.
- **Uso:** Análisis completo de la canción.

get_song_stats_only(song_title: str)

- **Parámetro:**
 - song_title: título de la canción.
- **Respuesta:**
 - Estadísticas rápidas: total de palabras, únicas, número de líneas, densidad léxica, tendencia emocional e intensidad emocional.
- **Uso:** Obtener un resumen breve sin análisis profundo.

2. Remoto

- **Nombre:** Weather
- **Transporte:** sse

Herramientas

get_alerts(state: str)

- **Parámetro:**
 - state: código de dos letras de un estado de EE. UU. (ej. CA, NY).
- **Respuesta:**
 - Lista de alertas activas formateadas con:
 - Evento
 - Área afectada
 - Severidad
 - Descripción
 - Instrucciones
- **Uso:** Consultar alertas meteorológicas actuales por estado.

get_forecast(latitude: float, longitude: float)

- **Parámetros:**
 - latitude: latitud de la ubicación.
 - longitude: longitud de la ubicación.

```

238 2025-09-25 18:39:12.930901 192.168.1.64 34.201.103.76 TCP
239 2025-09-25 18:39:12.931178 192.168.1.64 34.201.103.76 HTTP/JSON
240 2025-09-25 18:39:13.007372 34.201.103.76 192.168.1.64 TCP
241 2025-09-25 18:39:13.007455 34.201.103.76 192.168.1.64 TCP

Frame 239: 206 bytes on wire (1648 bits), 206 bytes captured (1648 bits)
Ethernet II, Src: Intel_a4:62:2b (4c:0c:4f:a4:62:2b), Dst: TIGRO_04:28:cd (74:
Internet Protocol Version 4, Src: 192.168.1.64, Dst: 34.201.103.76
Transmission Control Protocol, Src Port: 52568, Dst Port: 8000, Seq: 293, Ack:
[ 2 Reassembled TCP Segments (444 bytes): #238(292), #239(152)]
Hypertext Transfer Protocol
JavaScript Object Notation: application/json
  Object
    Member: method
      [Path with value: /method:initialize]
      [Member with value: method:initialize]
      String value: initialize
      Key: method
      [Path: /method]
    Member: params
      Object
        Member: protocolVersion:
          [Path with value: /params/protocolVersion:2025-06-18]
          [Member with value: protocolVersion:2025-06-18]
          String value: 2025-06-18

```

Este registro indica que la solicitud fue aceptada

| | | | | | | |
|---|----------------------------|---------------|---------------|-----------|--|---|
| 239 | 2025-09-25 18:39:12.931178 | 192.168.1.64 | 34.201.103.76 | HTTP/JSON | POST /messages/?session_id=cfa8bcbfa824012b5874a4b90eff2f1 H.. | 8000 → 52568 [ACK] Seq=1 Ack=293 Win=62464 Len=0 |
| 240 | 2025-09-25 18:39:13.007372 | 34.201.103.76 | 192.168.1.64 | TCP | | 8000 → 52568 [ACK] Seq=1 Ack=445 Win=62336 Len=0 |
| 241 | 2025-09-25 18:39:13.007455 | 34.201.103.76 | 192.168.1.64 | TCP | | 8000 → 52568 [PSH, ACK] Seq=1 Ack=445 Win=62336 Len=98 [TCP P.. |
| 242 | 2025-09-25 18:39:13.011904 | 34.201.103.76 | 192.168.1.64 | HTTP | HTTP/1.1 202 Accepted | |
| 243 | 2025-09-25 18:39:13.011904 | 34.201.103.76 | 192.168.1.64 | SSH | Server: Encrypted packet (len=208) | |
| 246 | 2025-09-25 18:39:13.017686 | 34.201.103.76 | 192.168.1.64 | TCP | | 8000 → 52567 [PSH, ACK] Seq=309 Ack=189 Win=62592 Len=298 [TC.. |
| 247 | 2025-09-25 18:39:13.019107 | 34.201.103.76 | 192.168.1.64 | TCP | | 52568 → 8000 [PSH, ACK] Seq=445 Ack=107 Win=65280 Len=291 [TC.. |
| 248 | 2025-09-25 18:39:13.023710 | 192.168.1.64 | 34.201.103.76 | HTTP/JSON | POST /messages/?session_id=cfa8bcbfa824012b5874a4b90eff2f1 H.. | 52567 → 8000 [ACK] Seq=189 Ack=607 Win=64768 Len=0 |
| 249 | 2025-09-25 18:39:13.023934 | 192.168.1.64 | 34.201.103.76 | TCP | | 61791 → 22 [ACK] Seq=129 Ack=1409 Win=255 Len=0 |
| 250 | 2025-09-25 18:39:13.062708 | 192.168.1.64 | 34.201.103.76 | TCP | | |
| 251 | 2025-09-25 18:39:13.062708 | 192.168.1.64 | 34.201.103.76 | TCP | | |
| Frame 243: 62 bytes on wire (496 bits), 62 bytes captured (496 bits) on 0 | | | | | | 0000 48 54 54 50 2f 31 2e 31 20 32 30 32 20 41 63 63 HTTP/1.1 202 Acc |
| Ethernet II, Src: TI80.04:28:cd (74:24:9f:04:28:cd), Dst: Intel_a4:62:2b (4c:00:1 | | | | | | 0010 65 70 74 65 64 0d 0a 64 61 74 65 3a 20 46 72 69 epted: d ate: Fri |
| Internet Protocol Version 4, Src: 34.201.103.76, Dst: 192.168.1.64 | | | | | | 0020 2c 20 32 36 20 53 65 70 20 32 30 32 35 20 30 30 , 26 Sep 2025 00 |
| Transmission Control Protocol, Src Port: 8000, Dst Port: 52568, Seq: 99, Ack: 4 | | | | | | 0030 3a 33 39 3a 31 33 20 47 4d 54 0d 0a 73 65 72 76 :39:13 GMT+serv |
| [2 Reassembled TCP Segments (106 bytes): #242(98), #243(8)] | | | | | | 0040 65 72 3a 20 75 76 69 63 6f 72 6e 0d 0a 63 6f 6e er: uvic orn+con |
| Hypertext Transfer Protocol | | | | | | 0050 74 65 6e 74 2d 6c 65 6e 67 74 68 3a 20 38 0d 0a tent-len gth: 8... |
| HTTP/1.1 202 Accepted\r\n | | | | | | 0060 0d 0a 41 63 63 65 70 74 65 64 |
| Response Version: HTTP/1.1 | | | | | | |
| Status Code: 202 | | | | | | |
| [Status Code Description: Accepted] | | | | | | |
| Response Phrase: Accepted | | | | | | |
| date: Fri, 26 Sep 2025 00:39:13 GMT\r\n | | | | | | |
| server: uvicorn\r\n | | | | | | |
| content-length: 8\r\n | | | | | | |
| \r\n | | | | | | |
| [Request in frame: 239] | | | | | | |
| [Time since request: 0.080726000 seconds] | | | | | | |

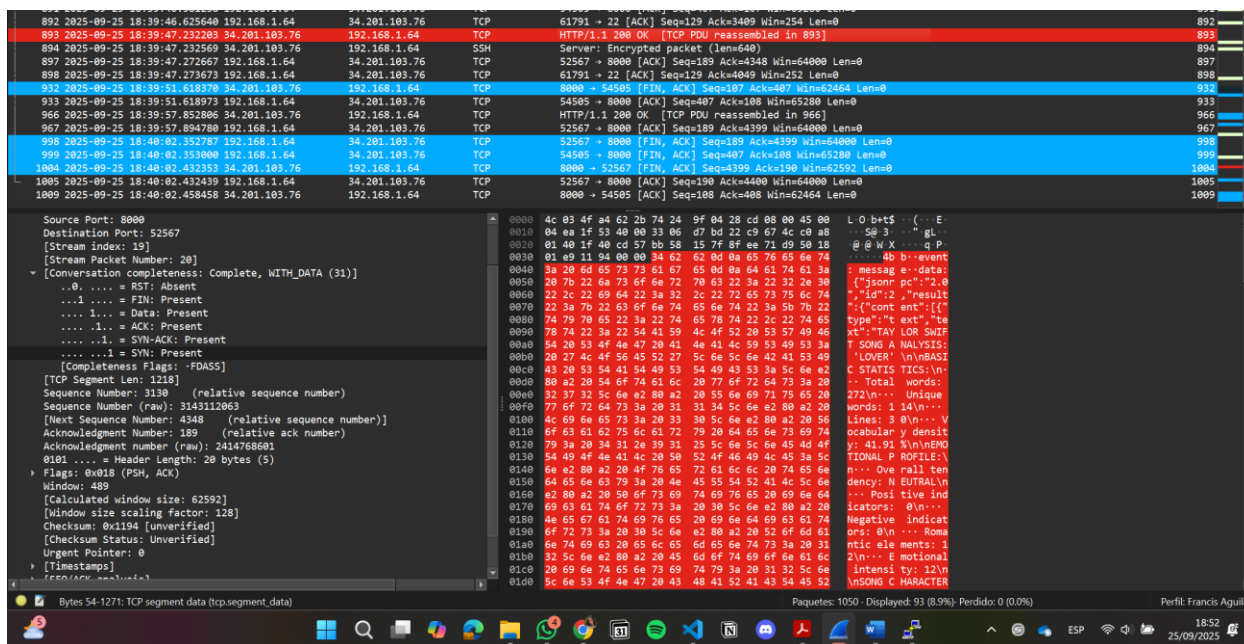
Aquí se muestra como se mandan a llamar el tool list

| | | | | | |
|---|----------------------------|---------------|---------------|-----------|---|
| 254 | 2025-09-25 18:39:13.104214 | 34.201.103.76 | 192.168.1.64 | TCP | 8000 → 52568 [PSH, ACK] Seq=107 Ack=790 Win=65280 Len=98 [TCP.. |
| 255 | 2025-09-25 18:39:13.104214 | 34.201.103.76 | 192.168.1.64 | HTTP | HTTP/1.1 202 Accepted |
| 256 | 2025-09-25 18:39:13.108652 | 192.168.1.64 | 34.201.103.76 | TCP | 52568 → 8000 [PSH, ACK] Seq=790 Ack=213 Win=65280 Len=291 [TC.. |
| 257 | 2025-09-25 18:39:13.108854 | 192.168.1.64 | 34.201.103.76 | HTTP/JSON | POST /messages/?session_id=cfa8bcbfa824012b5874a4b90eff2f1 H.. |
| 258 | 2025-09-25 18:39:13.157943 | 192.168.1.64 | 34.201.103.76 | TCP | 61791 → 22 [ACK] Seq=129 Ack=1617 Win=255 Len=0 |
| 259 | 2025-09-25 18:39:13.159515 | 34.201.103.76 | 192.168.1.64 | TCP | 8000 → 52568 [ACK] Seq=213 Ack=1127 Win=61824 Len=0 |
| 260 | 2025-09-25 18:39:13.159515 | 34.201.103.76 | 192.168.1.64 | SSH | Server: Encrypted packet (len=208) |
| 261 | 2025-09-25 18:39:13.204153 | 34.201.103.76 | 192.168.1.64 | TCP | 8000 → 52568 [PSH, ACK] Seq=213 Ack=1127 Win=61824 Len=98 [TC.. |
| 262 | 2025-09-25 18:39:13.204153 | 34.201.103.76 | 192.168.1.64 | SSH | Server: Encrypted packet (len=518) |
| 263 | 2025-09-25 18:39:13.204280 | 192.168.1.64 | 34.201.103.76 | TCP | 61791 → 22 [ACK] Seq=129 Ack=2353 Win=252 Len=0 |
| Frame 257: 100 bytes on wire (800 bits), 100 bytes captured (800 bits) on 0 | | | | | |
| Ethernet II, Src: Intel_a4:62:2b (4c:03:4f:a4:62:2b), Dst: TI80.04:28:cd (74:24 | | | | 0000 | 50 4f 53 54 20 2f 6d 65 73 73 61 67 65 73 2f 3f POST /me ssages/? |
| Internet Protocol Version 4, Src: 192.168.1.64, Dst: 34.201.103.76 | | | | 0010 | 73 65 73 73 69 6f 6e 5f 69 64 3d 63 66 30 61 38 session_id=cfa8 |
| Transmission Control Protocol, Src Port: 52568, Dst Port: 8000, Seq: 1081, Ack: | | | | 0020 | 62 63 30 66 61 38 32 3a 30 31 32 62 35 38 37 3a bcbfa824 012b5874 |
| [2 Reassembled TCP Segments (337 bytes): #256(291), #257(46)] | | | | 0030 | 2f 31 2e 31 0d 0a 68 6f 73 74 3a 20 33 34 2e 32 a4b90eff 2f1 HTTP |
| Hypertext Transfer Protocol | | | | 0040 | 30 31 2e 31 30 33 2e 37 36 3a 38 30 30 30 0d 0a /1.1 Ho st: 34.2 |
| JavaScript Object Notation: application/json | | | | 0050 | 41 63 63 65 70 74 2d 45 6e 63 6f 64 69 6e 67 3a 01.103.7 6:8000 |
| Object | | | | 0060 | 20 67 7a 69 70 2c 20 64 65 66 6c 61 74 65 0d 0a Accept-encoding: |
| Member: method | | | | 0070 | 43 6f 6e 6e 65 63 74 69 6f 6e 6e 3a 20 6b 65 65 70 gzip, deflate |
| [Path with value: /method:tools/list] | | | | 0080 | 2d 61 6c 69 76 65 0d 0a 55 73 65 72 2d 41 67 65 nt: pyth on-http |
| [Member with value: method:tools/list] | | | | 0090 | 61 3a 62 30 70 65 66 6e 32 66 31 30 48 54 50 61 68 6f 73 74 3a 20 33 34 2e 32 |
| String value: tools/list | | | | 00a0 | 2f 30 2e 32 38 2e 31 0d 0a 41 63 63 65 70 74 3a /0.28.1. Accept: |
| Key: method | | | | 00b0 | 20 74 65 78 74 2f 65 76 65 6e 74 2d 73 74 72 65 text/event-stre |
| [Path: /method] | | | | 00c0 | 61 6d 0d 0a 43 61 63 68 65 2d 43 6f 6e 74 72 6f am Cach e-Contro |
| Member: jsonrpc | | | | 00d0 | 6c 3a 20 6e 6f 2d 63 61 63 68 65 0d 0a 43 6f 6e 1: no-ca che Con |
| [Path with value: /jsonrpc:2.0] | | | | 00e0 | 74 65 6e 74 2d 4c 65 6e 67 74 68 3a 20 34 36 0d tent-len gth: 46 |
| [Member with value: jsonrpc:2.0] | | | | 00f0 | 0a 43 6f 6e 74 65 6e 74 2d 54 79 70 65 3a 20 61 Content -Type: a |
| String value: 2.0 | | | | 0100 | 70 70 6c 69 63 61 74 69 6f 6e 2f 6a 73 6f 6e 0d pplicati on/json |
| Key: jsonrpc | | | | 0110 | 0a 0d 0a 7b 22 6d 65 74 68 6f 64 22 3a 22 74 6f {"meth od": "to |
| [Path: /jsonrpc] | | | | 0120 | 6f 6c 73 2f 6c 69 73 74 22 2c 22 6a 73 6f 6e 72 ols/list ", "jsonr |
| Member: id | | | | 0130 | 70 63 22 3a 22 32 2e 30 22 2c 22 69 64 22 3a 31 pc": "2.0 ", "id": 1 |
| [Path with value: /id:1] | | | | 0140 | |
| [Member with value: id:1] | | | | 0150 | 7d |
| Number value: 1 | | | | | |
| Key: id | | | | | |
| [Path: /id] | | | | | |

Luego cuando se hace la consulta, aquí se puede observar que manda a llamar a analyze_song, una de las herramientas del mcp desarrollado

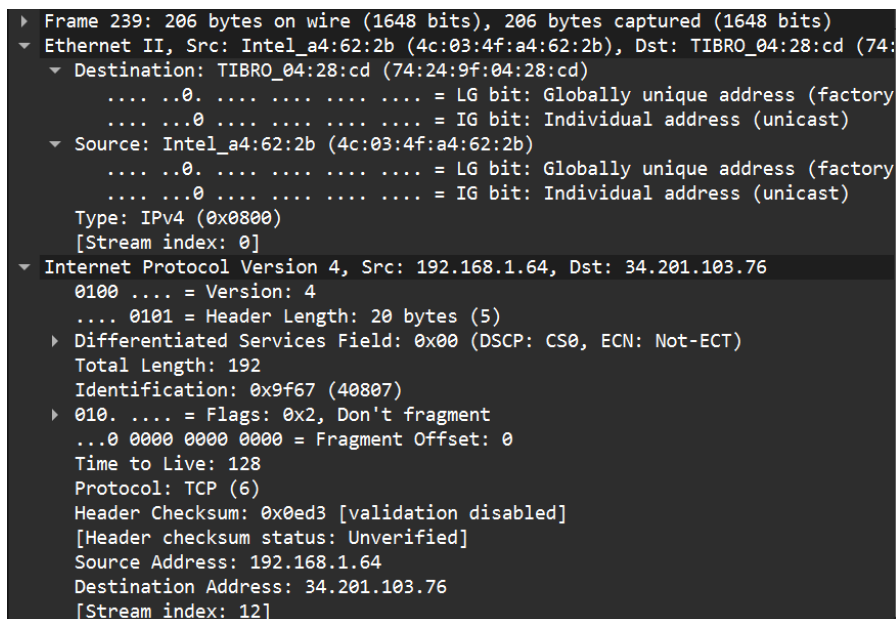
| | | | | | | |
|---|----------------------------|---------------|---------------|-----------|---|---|
| 881 | 2025-09-25 18:39:46.480870 | 192.168.1.64 | 34.201.103.76 | TCP | 54505 → 8000 [ACK] Seq=1 Ack=1 Win=65280 Len=0 | 881 |
| 882 | 2025-09-25 18:39:46.490410 | 192.168.1.64 | 34.201.103.76 | TCP | 54505 → 8000 [PSH, ACK] Seq=1 Ack=1 Win=65280 Len=292 [TCP PD.. | 882 |
| 883 | 2025-09-25 18:39:46.490778 | 192.168.1.64 | 34.201.103.76 | HTTP/JSON | POST /messages/?session_id=cfa8bcbfa824012b5874a4b90eff2f1 H.. | 883 |
| 884 | 2025-09-25 18:39:46.580264 | 34.201.103.76 | 192.168.1.64 | SSH | Server: Encrypted packet (len=208) | 884 |
| 885 | 2025-09-25 18:39:46.580264 | 34.201.103.76 | 192.168.1.64 | SSH | Server: Encrypted packet (len=528) | 885 |
| 886 | 2025-09-25 18:39:46.580363 | 192.168.1.64 | 34.201.103.76 | TCP | 61791 → 22 [ACK] Seq=129 Ack=1889 Win=255 Len=0 | 886 |
| 887 | 2025-09-25 18:39:46.580527 | 34.201.103.76 | 192.168.1.64 | SSH | Server: Encrypted packet (len=320) | 887 |
| 888 | 2025-09-25 18:39:46.580527 | 34.201.103.76 | 192.168.1.64 | TCP | 8000 → 54505 [ACK] Seq=1 Ack=407 Win=62464 Len=0 | 888 |
| 889 | 2025-09-25 18:39:46.580527 | 34.201.103.76 | 192.168.1.64 | TCP | 8000 → 54505 [PSH, ACK] Seq=1 Ack=407 Win=62464 Len=98 [TCP P.. | 889 |
| 890 | 2025-09-25 18:39:46.581143 | 34.201.103.76 | 192.168.1.64 | HTTP | HTTP/1.1 202 Accepted | 890 |
| Frame 889: 100 bytes on wire (800 bits), 100 bytes captured (800 bits) on 0 | | | | | | 0000 50 4f 53 54 20 2f 6d 65 73 73 61 67 65 73 2f 3f POST /me ssages/? |
| Key: method | | | | | | 0010 73 65 73 73 69 6f 6e 5f 69 64 3d 63 66 30 61 38 session_id=cfa8 |
| [Path: /method] | | | | | | 0020 62 63 30 66 61 38 32 3a 30 31 32 62 35 38 37 3a bcbfa824 012b5874 |
| Member: params | | | | | | 0030 61 34 62 39 30 65 66 66 66 32 66 31 20 48 54 54 50 a4b90eff 2f1 HTTP |
| Object | | | | | | 0040 2f 31 2e 31 30 33 2e 37 36 3a 38 30 30 30 0d 0a /1.1 Ho st: 34.2 |
| Member: name | | | | | | 0050 30 31 2e 31 30 33 2e 37 36 3a 38 30 30 30 0d 0a 01.103.7 6:8000 |
| [Path with value: /params/name:analyze_song] | | | | | | 0060 41 63 63 65 70 74 2d 45 6e 63 6f 64 69 6e 67 3a Accept-encoding: |
| [Member with value: name:analyze_song] | | | | | | 0070 20 67 7a 69 70 2c 20 64 65 66 6c 61 74 65 0d 0a gzip, deflate |
| String value: analyze_song | | | | | | 0080 43 6f 6e 6e 65 63 74 69 6f 6e 6e 3a 20 6b 65 65 70 nt: pyth on-http |
| Key: name | | | | | | 0090 2d 61 6c 69 76 65 0d 0a 55 73 65 72 2d 41 67 65 -alive User-Age |
| [Path: /params/name] | | | | | | 00a0 6e 74 3a 20 70 79 74 68 6f 6e 2d 68 74 70 78 70 /0.28.1. Accept: |
| Member: arguments | | | | | | 00b0 2f 30 2e 32 38 2e 31 0d 0a 41 63 63 65 70 74 3a text/event-stre |
| Object | | | | | | 00c0 20 74 65 78 74 2f 65 76 65 6e 74 2d 73 74 72 65 am Cach e-Contro |
| Member: song_title | | | | | | 00d0 6c 3a 20 6e 6f 2d 63 61 63 68 65 2d 43 6f 6e 74 72 6f 1: no-ca che Con |
| [Path with value: /params/arguments/song_title:Lover] | | | | | | 00e0 74 65 6e 74 2d 4c 65 6e 67 74 68 3a 20 31 31 34 Content-t: type: |
| [Member with value: song_title:Lover] | | | | | | 00f0 0d 0a 43 6f 6e 74 65 6e 74 2d 54 79 70 65 3a 20 tent-len gth: 114 |
| String value: Lover | | | | | | 0100 61 70 6c 69 63 61 74 69 6f 6e 2f 6a 73 6f 6e 0d applicat ion/json |
| Key: song_title | | | | | | 0110 0d 0a 0d 0a 7b 22 6d 65 74 68 6f 64 22 3a 22 74 6f {"meth od": "t |
| [Path: /params/arguments/song_title] | | | | | | 0120 6f 6c 73 2f 6c 69 73 74 22 2c 22 6a 73 6f 6e 72 ols/cal l", "para |
| Key: arguments | | | | | | 0130 70 63 22 3a 22 32 2e 30 22 2c 22 69 64 22 3a 31 me": "me me", "na |
| [Path: /params/arguments] | | | | | | 0140 6c 79 74 65 5f 73 6f 6e 67 22 2c 22 61 72 75 nze_son e", "argu |
| Key: params | | | | | | 0150 6d 65 6e 74 73 22 3a 7b 22 73 6f 6e 6f 5f 74 69 ments": { "song_ti |
| [Path: /params] | | | | | | 0160 74 6c 22 22 22 6f 76 65 72 7d 70 2c 22 74 6c 22 tle": "lo ver", "j |
| Member: jsonrpc | | | | | | 0170 6a 73 6f 6e 72 70 63 22 3a 22 32 2e 30 22 2c 22 jsonrpc": "2.0 ", |
| [Path with value: /jsonrpc:2.0] | | | | | | 0180 69 64 22 3a 32 |
| [Member with value: jsonrpc:2.0] | | | | | | |
| String value: 2.0 | | | | | | |

Y por último aquí se muestra la respuesta del mcp remoto con lo que tiene que devolver.



Capa de enlace de datos

En esta capa observamos la comunicación directa entre dos dispositivos físicos conectados en la misma red local. La trama Ethernet muestra que un dispositivo con tarjeta de red Intel (MAC: 4c:03:4f:a4:62:2b) está enviando datos hacia un dispositivo (MAC: 74:24:9f:04:28:cd)



Capa de red

El header IPv4 revela un paquete originado desde la red privada 192.168.1.64 dirigido hacia un servidor en la nube AWS (34.201.103.76). Podemos ver que se expone en el puerto 8000 tambien.

Capa de transporte

La información TCP revela una conexión establecida y madura en proceso de transmisión de datos finales. El puerto origen 52568 es un puerto efímero asignado dinámicamente al cliente, mientras que el puerto destino 8000 sugiere un servidor de aplicación personalizado (no HTTP estándar)

```
[Stream index: 12]
▼ Transmission Control Protocol, Src Port: 52568, Dst Port: 8000, Seq: 293
  Source Port: 52568
  Destination Port: 8000
  [Stream index: 20]
  [Stream Packet Number: 5]
  ▼ [Conversation completeness: Complete, WITH_DATA (31)]
    ..0. .... = RST: Absent
    ...1 .... = FIN: Present
    .... 1... = Data: Present
    .... .1.. = ACK: Present
    .... ..1. = SYN-ACK: Present
    .... ...1 = SYN: Present
    [Completeness Flags: ·FDASS]
  [TCP Segment Len: 152]
  Sequence Number: 293      (relative sequence number)
  Sequence Number (raw): 1831539073
  [Next Sequence Number: 445      (relative sequence number)]
  Acknowledgment Number: 1      (relative ack number)
  Acknowledgment number (raw): 1932870162
  0101 .... = Header Length: 20 bytes (5)
  ► Flags: 0x018 (PSH, ACK)
```

Capa aplicación

El patrón de comunicación desde una IP privada hacia un servidor AWS, combinado con el cierre ordenado de la conexión, es típico de aplicaciones web modernas que realizan llamadas a servicios en la nube.

Conclusiones

- El desarrollo de versiones local y remota del MCP me permitió entender cómo funciona la comunicación entre servidores y clientes, tanto en un entorno controlado (local) como en uno distribuido (remoto).
- Crear una versión remota y otra local me permitió comparar las ventajas y limitaciones de cada enfoque. Aprendí que la versión local es ideal para pruebas rápidas y control del sistema, mientras que la remota abre posibilidades de escalabilidad, colaboración y acceso desde distintos dispositivos.
- El proyecto me ayudó a aplicar conocimientos de despliegue en servidores, manejo de dependencias y configuración de entornos, lo que son habilidades directamente transferibles a proyectos profesionales.
- Haber construido mi propio MCP y cliente me dio independencia de soluciones cerradas, mostrándome que es posible crear herramientas a la medida, con más control sobre la seguridad, personalización y manejo de datos.