

Implementación de un protocolo ya existente

Gerardo Pineda 22880

Repositorios

Cliente y servidor remoto: [Gerax5/Redes_Proyecto1](#)

Servidor local: [Gerax5/MCP_Server](#)

MCPs

Local:

El MCP local implementado permite consultar sobre como es la comunidad de un canal de youtube. Este mcp extrae con la ayuda del api de youtube comentarios de videos y a estos les aplica un análisis de sentimientos para conocer mas a su comunidad. También, esta manda las palabras con más frecuencia. La herramienta es:

youtube_analysis:

- **descripción:**
 - Analiza la comunidad, audiencia, seguidores y comentarios de canales de YouTube para entender su demografía y comportamiento
- **parámetros:**
 - Channel* (Esta es obligatoria): El nombre del canal de youtube
 - max_comments: cuantos comentarios solicita.
 - top_words: Cuantas palabras con mayor frecuencia solicita.

Remoto:

El mcp remoto realiza una llamada a la api de Pokémon para obtener las estadísticas del Pokémon solicitado. La herramienta es:

get_pokemon

- **descripción:**
 - Permite consultar estadísticas, tipos y otra información detallada de un Pokémon
- **Parámetros:**
 - Name: Nombre del Pokémon.

Analisis de Wireshark

Desgloce de lo que se ve en Wireshack

Estos fueron los registros que se encontraron en una simple interacción entre el chatbot y el mcp remoto.

No.	Time	Source	Destination	Protocol	Info	New Column
59	3.364443	192.168.71.63	44.244.89.6	HTTP/J...	POST /mcp HTTP/1.1 , JSON (applica...	
71	3.567661	44.244.89.6	192.168.71.63	HTTP/J...	HTTP/1.1 200 OK , JSON (applicatio...	
74	3.568867	192.168.71.63	44.244.89.6	HTTP/J...	POST /mcp HTTP/1.1 , JSON (applica...	
97	3.745029	44.244.89.6	192.168.71.63	HTTP/J...	HTTP/1.1 200 OK , JSON (applicatio...	
113	3.841562	192.168.71.63	44.244.89.6	HTTP/J...	POST /mcp HTTP/1.1 , JSON (applica...	
123	4.012866	44.244.89.6	192.168.71.63	HTTP/J...	HTTP/1.1 200 OK , JSON (applicatio...	
282	9.752841	192.168.71.63	44.244.89.6	HTTP/J...	POST /mcp HTTP/1.1 , JSON (applica...	
290	10.062567	44.244.89.6	192.168.71.63	HTTP/J...	HTTP/1.1 200 OK , JSON (applicatio...	

Inicializacion

Primero el cliente manda al endpoint /mcp el método inicializar

No.	Time	Source	Destination	Protocol	Info	New Column
59	3.364443	192.168.71.63	44.244.89.6	HTTP/J...	POST /mcp HTTP/1.1 , JSON (applica...	
71	3.567661	44.244.89.6	192.168.71.63	HTTP/J...	HTTP/1.1 200 OK , JSON (applicatio...	
74	3.568867	192.168.71.63	44.244.89.6	HTTP/J...	POST /mcp HTTP/1.1 , JSON (applica...	
97	3.745029	44.244.89.6	192.168.71.63	HTTP/J...	HTTP/1.1 200 OK , JSON (applicatio...	
113	3.841562	192.168.71.63	44.244.89.6	HTTP/J...	POST /mcp HTTP/1.1 , JSON (applica...	
123	4.012866	44.244.89.6	192.168.71.63	HTTP/J...	HTTP/1.1 200 OK , JSON (applicatio...	
282	9.752841	192.168.71.63	44.244.89.6	HTTP/J...	POST /mcp HTTP/1.1 , JSON (applica...	
290	10.062567	44.244.89.6	192.168.71.63	HTTP/J...	HTTP/1.1 200 OK , JSON (applicatio...	


```

Frame 59: 415 bytes on wire (3320 bits), 415 bytes captured (3320 bits) on interface \Device\NPF...
Ethernet II, Src: Intel_e2:ad:a1 (8c:f8:c5:e2:ad:a1), Dst: Fortinet_09:00:1c (00:09:0f:09:00:1c)
Internet Protocol Version 4, Src: 192.168.71.63, Dst: 44.244.89.6
Transmission Control Protocol, Src Port: 61049, Dst Port: 8080, Seq: 1, Ack: 1, Len: 361
Hypertext Transfer Protocol
  JavaScript Object Notation: application/json
    Object
      Member: jsonrpc
      Member: id
      Member: method
        [Path with value: /method:initialize]
        [Member with value: /method:initialize]
        String value: initialize
        Key: method
        [Path: /method]
      Member: params
  
```

El servidor le responde con un 200 ok y le manda la versión del protocolo y capabilities.

No.	Time	Source	Destination	Protocol	Info	New Column
59	3.364443	192.168.71.63	44.244.89.6	HTTP/J...	POST /mcp HTTP/1.1 , JSON (applica...	
71	3.567661	44.244.89.6	192.168.71.63	HTTP/J...	HTTP/1.1 200 OK , JSON (applicatio...	
74	3.568867	192.168.71.63	44.244.89.6	HTTP/J...	POST /mcp HTTP/1.1 , JSON (applica...	
97	3.745029	44.244.89.6	192.168.71.63	HTTP/J...	HTTP/1.1 200 OK , JSON (applicatio...	
113	3.841562	192.168.71.63	44.244.89.6	HTTP/J...	POST /mcp HTTP/1.1 , JSON (applica...	
123	4.012866	44.244.89.6	192.168.71.63	HTTP/J...	HTTP/1.1 200 OK , JSON (applicatio...	
282	9.752841	192.168.71.63	44.244.89.6	HTTP/J...	POST /mcp HTTP/1.1 , JSON (applica...	
290	10.062567	44.244.89.6	192.168.71.63	HTTP/J...	HTTP/1.1 200 OK , JSON (applicatio...	


```

String value: 2.0
Key: jsonrpc
[Path: /jsonrpc]
Member: id
[Path with value: /id:1]
[Member with value: id:1]
Number value: 1
Key: id
[Path: /id]
Member: result
  Object
    Member: capabilities
    Member: protocolVersion
      [Path with value: /result/protocolVersion:2024-11-05]
      [Member with value: protocolVersion:2024-11-05]
      String value: 2024-11-05
      Key: protocolVersion
      [Path: /result/protocolVersion]
      Key: result
      [Path: /result]
  
```

Luego el cliente le manda una notificación al mcp diciendo que ya se conectó y lo único que hace el servir es obtener este mensaje. En la imagen se puede ver como el servidor le contesta, pero esta interacción únicamente fue porque el servidor devuelve que el método

notification/initialized no esta implementado, pero en si debería de solo quedarse en el endpoint no devolver nada,

```
74 3.568867 192.168.71.63 44.244.89.6 HTTP/1.1 POST /mcp HTTP/1.1 , JSON (applicatio...
97 3.745629 44.244.89.6 192.168.71.63 HTTP/1.1 HTTP/1.1 200 OK , JSON (applicatio...
113 3.841562 192.168.71.63 44.244.89.6 HTTP/1.1 POST /mcp HTTP/1.1 , JSON (applicatio...
123 4.012866 44.244.89.6 192.168.71.63 HTTP/1.1 HTTP/1.1 200 OK , JSON (applicatio...
282 9.752841 192.168.71.63 44.244.89.6 HTTP/1.1 POST /mcp HTTP/1.1 , JSON (applicatio...
290 10.062567 44.244.89.6 192.168.71.63 HTTP/1.1 HTTP/1.1 200 OK , JSON (applicatio...

> Transmission Control Protocol, Src Port: 61049, Dst Port: 8080, Seq: 362, Ack: 210, Len: 298
> Hypertext Transfer Protocol
> JavaScript Object Notation: application/json
  Object
    Member: jsonrpc
      [Path with value: /jsonrpc:2.0]
      [Member with value: jsonrpc:2.0]
      String value: 2.0
      Key: jsonrpc
      [Path: /jsonrpc]
    Member: method
      [Path with value: /method:notifications/initialized]
      [Member with value: method:notifications/initialized]
      String value: notifications/initialized
      Key: method
      [Path: /method]
    Member: params
      Object
      Key: params
      [Path: /params]
```

Luego el mcp pide el tool/list

```
113 3.841562 192.168.71.63 44.244.89.6 HTTP/1.1 POST /mcp HTTP/1.1 , JSON (applicatio...
123 4.012866 44.244.89.6 192.168.71.63 HTTP/1.1 HTTP/1.1 200 OK , JSON (applicatio...
282 9.752841 192.168.71.63 44.244.89.6 HTTP/1.1 POST /mcp HTTP/1.1 , JSON (applicatio...
290 10.062567 44.244.89.6 192.168.71.63 HTTP/1.1 HTTP/1.1 200 OK , JSON (applicatio...

> Member: jsonrpc
  [Path with value: /jsonrpc:2.0]
  [Member with value: jsonrpc:2.0]
  String value: 2.0
  Key: jsonrpc
  [Path: /jsonrpc]
> Member: id
  [Path with value: /id:2]
  [Member with value: id:2]
  Number value: 2
  Key: id
  [Path: /id]
> Member: method
  [Path with value: /method:tools/list]
  [Member with value: method:tools/list]
  String value: tools/list
  Key: method
  [Path: /method]
> Member: params
  Object
```

Y el servidor le responde las herramientas que tienen con los parámetros requeridos. Para que el cliente tenga noción de las herramientas que va a tener.

No.	Time	Source	Destination	Protocol	Info	New Column
59	3.364443	192.168.71.63	44.244.89.6	HTTP/3.	POST /mcp HTTP/1.1 , JSON (applicatio...	
71	3.567661	44.244.89.6	192.168.71.63	HTTP/3.	HTTP/1.1 200 OK , JSON (applicatio...	
74	3.568867	192.168.71.63	44.244.89.6	HTTP/3.	POST /mcp HTTP/1.1 , JSON (applicatio...	
97	3.745929	44.244.89.6	192.168.71.63	HTTP/3.	HTTP/1.1 200 OK , JSON (applicatio...	
113	3.841562	192.168.71.63	44.244.89.6	HTTP/3.	POST /mcp HTTP/1.1 , JSON (applicatio...	
123	4.012866	44.244.89.6	192.168.71.63	HTTP/3.	HTTP/1.1 200 OK , JSON (applicatio...	
282	9.752841	192.168.71.63	44.244.89.6	HTTP/3.	POST /mcp HTTP/1.1 , JSON (applicatio...	
290	10.062567	44.244.89.6	192.168.71.63	HTTP/3.	HTTP/1.1 200 OK , JSON (applicatio...	


```
Object
  Member: tools
    Array
      Object
        Member: name
          [Path with value: /result/tools/[]/name:get_pokemon]
          [Member with value: name:get_pokemon]
          String value: get_pokemon
          Key: name
          [Path: /result/tools/[]/name]
        Member: title
          [Path with value: /result/tools/[]/title:Pokémon Lookup]
          [Member with value: title:Pokémon Lookup]
          String value: Pokémon Lookup
          Key: title
          [Path: /result/tools/[]/title]
        Member: description
          [Path with value: /result/tools/[]/description:Permite consultar estadísticas, tipos y otra in]
          [Member with value: description:Permite consultar estadísticas, tipos y otra in]
          String value: Permite consultar estadísticas, tipos y otra información detallada...
```

Posteriormente el cliente solicita al servidor usar la herramienta get_pokemon, esta manda el nombre del pokemon

No.	Time	Source	Destination	Protocol	Info	New Column
59	3.364443	192.168.71.63	44.244.89.6	HTTP/3.	POST /mcp HTTP/1.1 , JSON (applicatio...	
71	3.567661	44.244.89.6	192.168.71.63	HTTP/3.	HTTP/1.1 200 OK , JSON (applicatio...	
74	3.568867	192.168.71.63	44.244.89.6	HTTP/3.	POST /mcp HTTP/1.1 , JSON (applicatio...	
97	3.745929	44.244.89.6	192.168.71.63	HTTP/3.	HTTP/1.1 200 OK , JSON (applicatio...	
113	3.841562	192.168.71.63	44.244.89.6	HTTP/3.	POST /mcp HTTP/1.1 , JSON (applicatio...	
123	4.012866	44.244.89.6	192.168.71.63	HTTP/3.	HTTP/1.1 200 OK , JSON (applicatio...	
282	9.752841	192.168.71.63	44.244.89.6	HTTP/3.	POST /mcp HTTP/1.1 , JSON (applicatio...	
290	10.062567	44.244.89.6	192.168.71.63	HTTP/3.	HTTP/1.1 200 OK , JSON (applicatio...	


```
Key: method
[Path: /method]
Member: params
  Object
    Member: name
      [Path with value: /params/name:get_pokemon]
      [Member with value: name:get_pokemon]
      String value: get_pokemon
    Key: name
      [Path: /params/name]
    Member: arguments
      Object
        Member: name
          [Path with value: /params/arguments/name:pikachu]
          [Member with value: name:pikachu]
          String value: pikachu
        Key: name
          [Path: /params/arguments/name]
        Key: arguments
          [Path: /params/arguments]
```

Por último el servidor responde lo obtenido

No.	Time	Source	Destination	Protocol	Info	New Column
59	3.364443	192.168.71.63	44.244.89.6	HTTP/3.	POST /mcp HTTP/1.1 , JSON (applic...	
71	3.567961	44.244.89.6	192.168.71.63	HTTP/3.	HTTP/1.1 200 OK , JSON (applicatio...	
74	3.568867	192.168.71.63	44.244.89.6	HTTP/3.	POST /mcp HTTP/1.1 , JSON (applic...	
97	3.745829	44.244.89.6	192.168.71.63	HTTP/3.	HTTP/1.1 200 OK , JSON (applicatio...	
113	3.841562	192.168.71.63	44.244.89.6	HTTP/3.	POST /mcp HTTP/1.1 , JSON (applic...	
123	4.012866	44.244.89.6	192.168.71.63	HTTP/3.	HTTP/1.1 200 OK , JSON (applicatio...	
282	9.752841	192.168.71.63	44.244.89.6	HTTP/3.	POST /mcp HTTP/1.1 , JSON (applic...	
298	10.862567	44.244.89.6	192.168.71.63	HTTP/3.	HTTP/1.1 200 OK , JSON (applicatio...	

Object
Member: content
Array
Object
Member: type
[Path with value: /result/content/[]/type:text]
Member with value: type:text
String value: text
Key: type
[Path: /result/content/[]/type]
Member: text
[Path with value: /result/content/[]/text:Pokémon Pikachu (ID 25): Tipos: electri
Member with value: text:Pokémon Pikachu (ID 25): Tipos: electric, Stats: {hp:
String value: Pokémon Pikachu (ID 25): Tipos: electric, Stats: {hp: 35, 'atta
Key: text
[Path: /result/content/[]/text]
Key: content
[Path: /result/content]
Key: result
[Path: /result]

0000 8c f8 c5 e2 ad a1 00 00 0f 09 00 1c 08 00 45 00E
0010 01 04 de 80 40 00 36 06 d7 91 2c f4 59 06 c0 a8 ...@6...Y...
0020 47 3f 1f 90 ee 7f 22 e2 14 1c 42 12 91 d5 50 18 G7...B...P
0030 01 e9 54 c1 00 80 7b 22 6a 73 6f 6e 72 70 63 22 ... ("jsonrpc"
0040 3a 22 32 2a 30 22 2c 22 69 64 22 3a 33 2c 22 72 ..."2.0","id":3,"r
0050 65 73 75 6c 74 22 3a 7b 22 63 6f 6e 74 65 6e 74 ...result":{"content
0060 22 3a 5b 7b 22 74 79 70 65 22 3a 22 74 65 78 74 ...":{"type":"text
0070 22 2c 22 74 65 78 74 22 3a 22 50 6f 6b c3 a9 64 ...","text":"Pok m
0080 6f 6e 20 50 69 6b 61 63 68 75 20 28 49 44 20 32 ...on Pikac hu (ID 2
0090 35 29 3a 20 54 69 70 6f 73 3a 20 65 6c 65 63 74 ...5): Tipo s: elect
00a0 72 69 63 2c 20 53 74 61 74 73 3a 20 7b 27 68 70 ...ric, Stats: {hp
00b0 27 3a 20 33 35 2c 20 27 61 74 74 61 63 6b 27 3a ...i 35, 'attack':
00c0 20 35 35 2c 20 27 64 65 66 65 6e 73 65 27 3a 20 ...55, 'de fense':
00d0 34 30 2c 20 27 73 70 65 63 69 61 6c 2d 61 74 74 ...40, 'spe cial-att
00e0 61 63 4b 27 3a 20 35 30 2c 20 27 73 70 65 63 69 ...ack': 50, 'speci
00f0 61 6c 2d 64 65 66 65 6e 73 65 27 3a 20 35 30 2c ...al-defen se': 50,
0100 20 27 73 70 65 65 64 27 3a 20 39 30 7d 22 7d 5d ...'speed' : 90")]}
0110 7d 7d

Capa de enlace

<ul style="list-style-type: none"> Ethernet II, Src: Intel_e2:ad:a1 (8c:f8:c5:e2:ad:a1), Dst: Fortinet_09:00:1c (00:09:0f:09:00:1c) <ul style="list-style-type: none"> Destination: Fortinet_09:00:1c (00:09:0f:09:00:1c) Source: Intel_e2:ad:a1 (8c:f8:c5:e2:ad:a1) Type: IPv4 (0x0800) [Stream index: 6]
--

Cada mensaje se envia por internet, pasa por el mac de origen que es la tarjeta de red y el destino que es el Gateway route.

Capa de red es mi fuente destino que se ven en las imágenes anteriores la ip 192.168.71.63 a la ip destino de aws 44.244.89.6 se usa un enrutamiento a través de red. La capa de transporte sucede en con TCP en el puerto 8080 del servidor y el puerto 61049 de la computadora

<ul style="list-style-type: none"> Transmission Control Protocol, Src Port: 61049, Dst Port: 8080, Seq: 1, Ack: 1, Len: 361 <ul style="list-style-type: none"> Source Port: 61049 Destination Port: 8080 [Stream index: 8] [Stream Packet Number: 4] [Conversation completeness: Complete, WITH_DATA (63)] [TCP Segment Len: 361] Sequence Number: 1 (relative sequence number) Sequence Number (raw): 1332624694 [Next Sequence Number: 362 (relative sequence number)] Acknowledgment Number: 1 (relative ack number) Acknowledgment number (raw): 92565095 0101 = Header Length: 20 bytes (5) Flags: 0x018 (PSH, ACK) Window: 255

Se puede observar cómo los ack y seq van aumentando en cada respuesta

<ul style="list-style-type: none"> Internet Protocol Version 4, Src: 192.168.71.63, Dst: 44.244.89.6 Transmission Control Protocol, Src Port: 61049, Dst Port: 8080, Seq: 1, Ack: 1, Len: 361 <ul style="list-style-type: none"> Source Port: 61049
<ul style="list-style-type: none"> Transmission Control Protocol, Src Port: 8080, Dst Port: 61049, Seq: 126, Ack: 362, Len: 84 <ul style="list-style-type: none"> Source Port: 8080

Y la capa de aplicación es la que mencione con anterioridad en el desglose de lo que mostraba wireshack con el protocolo http.

Dificultades

El apartado que representó mayor desafío fue lograr que el chatbot se comunicara correctamente con los servidores MCP. Mientras que implementar un chatbot con capacidad de mantener el contexto resultó relativamente sencillo, la verdadera complejidad apareció al integrar el uso de herramientas dentro de la comunicación con el protocolo. La dificultad se debió, en gran parte, a que era primera vez que usaba el lenguaje de programación y que no hay mucha documentación para implementar un chatbot con servidores mcp. Otra cosa que puedo mencionar, en mis MCP creados hice una implementación pura del protocolo sin recurrir a librerías como *fastmcp*. Finalmente, este reto se superó con recursos en línea y el análisis de ejemplos existentes, lo que permitió comprender mejor la estructura del protocolo e implementar con éxito la comunicación requerida.

Comentarios

Considero que es una buena forma de aplicar los conocimientos de la clase, agregándolo algo novedoso como usar un api de IA e integrarlo para hacer un propio chatbot. El cual pueda consultar a los servidores mcp. Buena forma de lograr hacer un servidor el cual pueda exponerse en la nube para que las demás personas la usen y consulten.

Conclusiones

- Los servidores MCP son útiles al integrarse con la inteligencia artificial, debido a que permiten automatizar procesos y reutilizar recursos que pueden ser desarrollador por nosotros mismos. Tendríamos una IA la cual es capaz de responder en base a su entrenamiento y herramientas externas que amplíen sus capacidades.
- Se logró implementar y analizar el protocolo MCP tanto en servidores locales como remotos, evidenciando cómo se establece la comunicación cliente-servidor a través de HTTP y confirmando la compatibilidad con los estándares definidos.
- Se evidenció el proceso de inicialización de la conexión, la notificación de estado, la consulta de herramientas disponibles y el uso de una herramienta específica. Cada una de estas etapas mostró el rol del protocolo en garantizar que el cliente entienda qué servicios puede consumir y cómo hacerlo.