

Reporte de Implementación MCP

CC3067 Redes — Proyecto 1
Uso de un protocolo existente

Autor: Gabriel Alberto Paz González 221087

Fecha: 25 de septiembre de 2025

Resumen

Este documento describe la implementación y evaluación de múltiples servidores MCP (Model Context Protocol) integrados en un host tipo chatbot. Se documentan las especificaciones, parámetros y endpoints de los servidores MCP desarrollados por el autor (SiteLens y RemoteMCP), así como el análisis de la comunicación remota capturada con Wireshark a nivel de las capas de enlace, red, transporte y aplicación. Finalmente, se presentan conclusiones y comentarios sobre el proyecto.

Especificación de los servidores MCP desarrollados

SiteLens (MCP local por STDIO)

Campo	Detalle
Descripción	Servidor MCP local (Node/TypeScript) que audita carpetas con HTML estático sin salir a la red.
Transporte	STDIO
Protocolo MCP	2024-11-05; métodos: initialize, tools/list, tools/call.
Control de acceso	Lista blanca de raíces vía --roots o variable ALLOWED_ROOTS (separadas por ';' en Windows).
Formato tabulable	{ "structuredContent": { "result": <lista obj> } }

Tools (SiteLens)

Tool	Parámetros	Descripción
aa.allowed_roots	{ }	Lista de roots efectivas.
aa.sitemap	{ path, includeHtmlOnly?, maxDepth? }	Árbol JSON del FS.
aa.link_check	{ path, entry?, extensions? }	Verifica enlaces internos (externos: skipped).
aa.asset_budget	{ path, patterns?, budgetKB? }	Totales por tipo, top pesados y sobre presupuesto.
aa.scan_accessibility	{ path, include?, exclude? }	Reglas WCAG-lite: alt, labels, landmarks, headings, contraste.
aa.report	{ path, weights?, top? }	Ranking (0–100) y quick wins.

Descripción: Servidor MCP local (Node/TypeScript) que audita carpetas con HTML estático sin red.

Transporte: STDIO.

Protocolo MCP: 2024-11-05; métodos soportados: initialize, tools/list, tools/call.

Control de acceso: whitelist de raíces vía --roots o variable ALLOWED_ROOTS (separadas por ';' en Windows).

Formato de respuesta tabulable: { "structuredContent": { "result": <lista|obj> } }.

- Tools expuestas:
 - aa.allowed_roots → {} → string[] de roots efectivas.
 - aa.sitemap → { path, includeHtmlOnly?, maxDepth? } → árbol JSON del FS.
 - aa.link_check → { path, entry?, extensions? } → validación de enlaces internos (externos: skipped).
 - aa.asset_budget → { path, patterns?, budgetKB? } → totales por tipo, top pesados y sobre presupuesto.
 - aa.scan_accessibility → { path, include?, exclude? } → reglas WCAG-lite: alt, labels, landmarks, headings, contraste.
 - aa.report → { path, weights?, top? } → ranking (0–100) y quick wins consolidando resultados previos.

Ejemplo de integración en el host (mcp_config.json):

```
{
  "name": "SiteLens",
  "transport": "stdio",
  "command": "node",
  "args": ["C:/UVG/PROYECTO1/MCP_Local/sitelens/dist/server.js", "--roots", "C:/UVG/PROYECTO1/MCP_Local/test/site"]
}
```

RemoteMCP (MCP remoto vía Cloudflare Workers)

RemoteMCP (Cloudflare Workers)

Campo	Detalle
Descripción	Servidor MCP remoto (Cloudflare Workers) con endpoints WSS y HTTP JSON-RPC 2.0.
Transportes	WebSocket seguro (WSS) y HTTP (POST).
Ruta de servicio	/mcp
Protocolo MCP	initialize, tools/list, tools/call.

Formato tabulable	{ "structuredContent": { "result": <obj> } }
-------------------	--

Endpoints

Tipo	URL
WSS	wss://remote-mcp-demo.gabouvg.workers.dev/mcp
HTTP	https://remote-mcp-demo.gabouvg.workers.dev/mcp

Tools (RemoteMCP)

Tool	Parámetros	Respuesta (ejemplo)
remote.ping	{}	{ "ok": true, "message": "pong" }
remote.time	{}	{ "now": "<ISO>" }
remote.echo	{ "text": "Hola" }	{ "echo": "Hola" }

Descripción: Servidor MCP remoto (Cloudflare Workers) con endpoints WSS y HTTP JSON-RPC 2.0.

Transporte: WebSocket seguro (WSS) y HTTP (POST).

Ruta de servicio: /mcp

Protocolo MCP: métodos initialize, tools/list, tools/call.

Formato de respuesta tabulable: { "structuredContent": { "result": <obj> } }.

- Endpoints:
 - WSS: wss://remote-mcp-demo.<subdominio>.workers.dev/mcp
 - HTTP: https://remote-mcp-demo.<subdominio>.workers.dev/mcp
- Tools expuestas (demo):
 - remote.ping → {} → { ok:true, message:"pong" }
 - remote.time → {} → { now:"<ISO>" }
 - remote.echo → { text:string } → { echo:"<text>" }

Ejemplo de integración en el host (mcp_config.json):

```
{
  "name": "RemoteMCP",
```

"transport": "websocket",
"url": "wss://remote-mcp-demo.<subdominio>.workers.dev/mcp"
}

Análisis por capas (basado en captura Wireshark del punto 8)

Capturando desde Wi-Fi

Archivo Edición Visualización Ir Captura Analizar Estadísticas Telefonía Wireless Herramientas Ayuda

[[http2 || http || https || https handshake.extensions_server_name contains "remote-mcp-demo.gabouvg.workers.dev"]]

No.	Time	Source	Destination	Protocol	Length	Info
2577	167.203231	172.20.10.3	172.67.188.200	TLSv1.3	571	Client Hello (SNI=remote-mcp-demo.gabouvg.workers.dev)
2629	169.168551	172.20.10.3	172.67.188.200	TLSv1.3	571	Client Hello (SNI=remote-mcp-demo.gabouvg.workers.dev)
2651	170.328229	172.20.10.3	172.67.188.200	TLSv1.3	571	Client Hello (SNI=remote-mcp-demo.gabouvg.workers.dev)
2686	171.373662	172.20.10.3	172.67.188.200	TLSv1.3	571	Client Hello (SNI=remote-mcp-demo.gabouvg.workers.dev)
2819	181.011798	172.20.10.3	172.67.188.200	TLSv1.3	571	Client Hello (SNI=remote-mcp-demo.gabouvg.workers.dev)
2841	181.749543	172.20.10.3	172.67.188.200	TLSv1.3	571	Client Hello (SNI=remote-mcp-demo.gabouvg.workers.dev)
2869	182.355987	172.20.10.3	172.67.188.200	TLSv1.3	571	Client Hello (SNI=remote-mcp-demo.gabouvg.workers.dev)
2894	183.259951	172.20.10.3	172.67.188.200	TLSv1.3	571	Client Hello (SNI=remote-mcp-demo.gabouvg.workers.dev)
2914	183.850824	172.20.10.3	172.67.188.200	TLSv1.3	571	Client Hello (SNI=remote-mcp-demo.gabouvg.workers.dev)
2938	184.316270	172.20.10.3	172.67.188.200	TLSv1.3	571	Client Hello (SNI=remote-mcp-demo.gabouvg.workers.dev)
2965	184.776528	172.20.10.3	172.67.188.200	TLSv1.3	571	Client Hello (SNI=remote-mcp-demo.gabouvg.workers.dev)
2987	185.388387	172.20.10.3	172.67.188.200	TLSv1.3	571	Client Hello (SNI=remote-mcp-demo.gabouvg.workers.dev)
3006	185.787940	172.20.10.3	172.67.188.200	TLSv1.3	571	Client Hello (SNI=remote-mcp-demo.gabouvg.workers.dev)

Frame 2651: 571 bytes on wire (4568 bits), 571 bytes captured (4568 bits) on interface \Device\NPF_{15A0...}

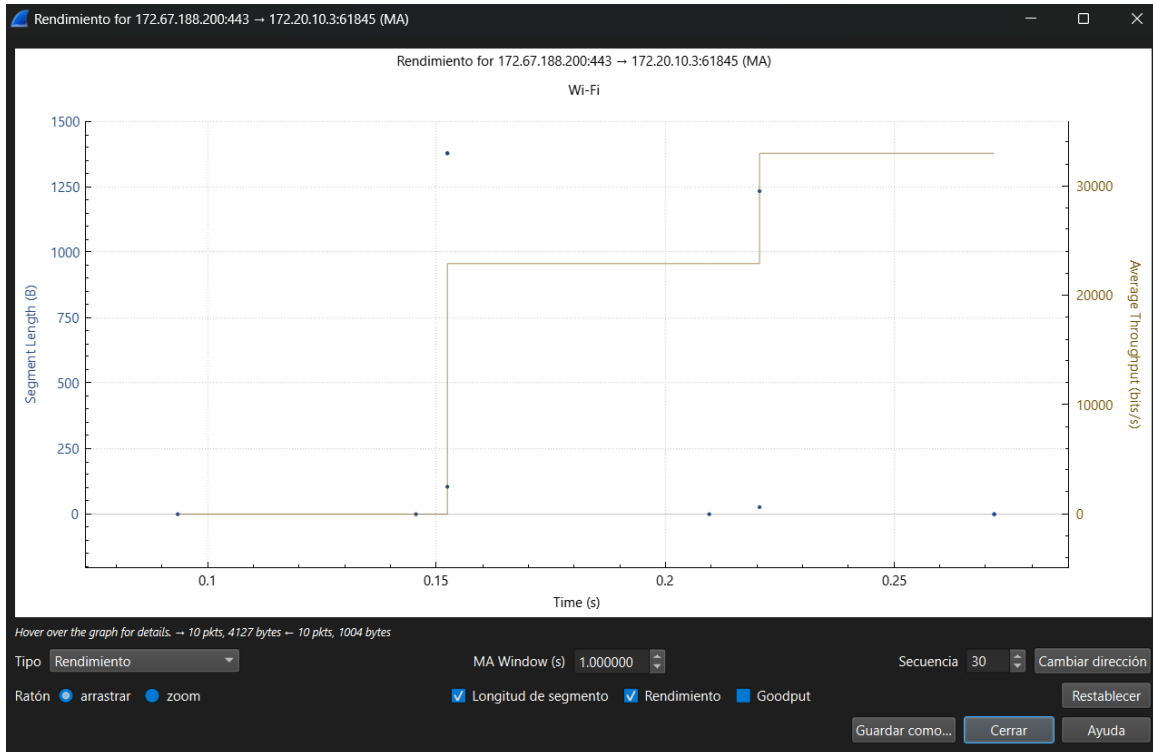
Ethernet II, Src: Intel_f6:65:a6:db:a9:64 (2c:6d:c1:fe:30:d0), Dst: f6:65:a6:db:a9:64 (f6:65:a6:db:a9:64)

Internet Protocol Version 4, Src: 172.20.10.3, Dst: 172.67.188.200

Transmission Control Protocol, Src Port: 61845, Dst Port: 443, Seq: 1, Ack: 1, Len: 517

Transport Layer Security

0000 f6 65 a6 db a9 64 2c 6d c1 fe 30 d0 00 00 45 00 e d m o E
0010 02 2d bf 74 40 00 00 06 00 00 ac 14 0a 03 ac 43 t g b C
0020 bc c8 f1 95 01 bb 09 2c b1 62 a8 b6 4d 2f 50 18 , b P/P
0030 00 ff 21 43 00 00 16 03 01 02 00 01 00 01 fc 03 IC
0040 03 06 f1 5d 7d 19 7b 6f 35 0a ae 8d dd 6e 87 1b }} {o 5
0050 30 90 f0 1d 22 e7 69 55 ab de ef 98 7f 71 85 e1 0 " 1u
0060 88 20 d7 06 f3 fa 9e 06 ad b5 90 11 a1 23 9e db
0070 a8 f9 9f fa 7a 75 d7 1d c6 27 cf 3d 58 f8 f1 b3 zu X
0080 50 be 00 24 13 02 13 03 13 01 c0 2c c0 30 c0 2b P \$, 0 +
0090 02 ff cc 49 cc a8 c0 24 c0 28 c0 23 c0 27 00 9f / - \$ (#
00a0 00 9a 00 05 00 67 00 ff 01 00 01 8f 00 00 00 28 k g
00b0 00 26 00 00 23 72 65 6d 6f 74 65 2d 6d 63 70 2d & #rem ote-mcp-
00c0 64 65 6d 6f 2e 67 61 62 6f 75 76 67 2e 77 6f 72 demo.gab ouvg.uoe
00d0 6b 65 72 73 2e 64 65 76 00 00 00 04 03 00 01 02 kers.dev
00e0 00 0a 00 16 00 14 00 1d 00 17 00 1e 00 19 00 18
00f0 01 01 01 01 01 02 01 03 01 04 00 10 00 0b 00 09
0100 08 68 74 74 70 2f 31 2e 31 00 16 00 00 00 17 00
0110 00 00 31 00 00 00 0d 00 2a 00 28 04 03 05 03 06
0120 03 08 07 08 08 08 09 08 0a 08 0b 08 04 08 05 08
0130 06 04 01 05 01 06 01 03 03 03 01 03 02 04 02 05
0140 02 06 02 00 2b 00 05 04 03 04 03 03 00 2d 00 02
0150 01 01 00 33 00 26 00 24 00 1d 00 20 93 13 1d f0 3 & \$



Source Address	Source Port	Destination Address	Destination Port	Paquetes	Packets/s	Avg BW (bps)	Max BW (bps)	Max Burst	Burst Alarms	Max Buffers (B)
fe80:1465:a6ff:fedba964	5353	#02:fb	5353	1	0.00	0	0	1 / 100ms	0	120
fe80:d27:e5bf:fe42bd118	5353	#02:fb	5353	2	0.17	188	0	1 / 100ms	0	142
fe80:9766:26a4:c6e1:9e5d	59225	#02:13	5355	2	4.74	3491	0	1 / 100ms	0	92
fe80:9766:26a4:c6e1:9e5d	56363	#02:13	5355	2	4.81	3542	0	1 / 100ms	0	92
fe80:9766:26a4:c6e1:9e5d	52754	#02:13	5355	2	4.83	3553	0	1 / 100ms	0	92
fe80:9766:26a4:c6e1:9e5d	57254	#02:13	5355	2	4.81	3539	0	1 / 100ms	0	92
fe80:9766:26a4:c6e1:9e5d	55780	#02:13	5355	2	4.86	3575	0	1 / 100ms	0	92
fe80:9766:26a4:c6e1:9e5d	49665	#02:13	5355	1	0.00	0	0	1 / 100ms	0	92
fe80:9766:26a4:c6e1:9e5d	60330	#02:13	5355	2	4.76	3501	0	1 / 100ms	0	92
fe80:9766:26a4:c6e1:9e5d	5353	#02:fb	5353	55	1.19	1128	24 k	3 / 100ms	0	102
fe80:9766:26a4:c6e1:9e5d	546	#02:12	547	3	0.99	1288	0	1 / 100ms	0	162
fe80:9766:26a4:c6e1:9e5d	59974	#02:13	5355	2	4.82	3548	0	1 / 100ms	0	92
fe80:9766:26a4:c6e1:9e5d	54025	#02:13	5355	2	4.80	3529	0	1 / 100ms	0	92
fe80:9766:26a4:c6e1:9e5d	53417	#02:13	5355	2	4.46	3279	0	1 / 100ms	0	92
fe80:9766:26a4:c6e1:9e5d	53588	#02:13	5355	2	4.85	3569	0	1 / 100ms	0	92
fe80:9766:26a4:c6e1:9e5d	59522	#02:13	5355	2	4.74	3487	0	1 / 100ms	0	92
fe80:9766:26a4:c6e1:9e5d	50593	#02:13	5355	2	4.77	3507	0	1 / 100ms	0	92
fe80:9766:26a4:c6e1:9e5d	51080	#02:13	5355	2	4.75	3493	0	1 / 100ms	0	92
fe80:9766:26a4:c6e1:9e5d	52372	#02:13	5355	2	4.84	3564	0	1 / 100ms	0	92
fe80:9766:26a4:c6e1:9e5d	49340	#02:13	5355	2	4.69	3448	0	1 / 100ms	0	92
fe80:9766:26a4:c6e1:9e5d	65313	#02:13	5355	2	4.80	3535	0	1 / 100ms	0	92
fe80:8f78:8fdc:462d:6052	5353	#02:fb	5353	22	0.17	521	103 k	3 / 100ms	0	866
fe80:101f2235:a5a0:801b	5353	#02:fb	5353	3	0.74	706	0	1 / 100ms	0	120
fe80:c9b2becaa426:54ca	5353	#02:fb	5353	4	0.03	34	0	1 / 100ms	0	142
fe80:878:410a:2d1f8:409d	5353	#02:fb	5353	6	0.21	205	0	1 / 100ms	0	120

51 streams, avg bw: 118bps, max bw: 57 kbps, max burst: 7 / 100ms, max buffer: 1313 MB

Burst measurement interval (ms): 100
Burst alarm threshold (packets): 50
Buffer alarm threshold (B): 10000

Stream empty speed (Kb/s): 5000
Total empty speed (Kb/s): 100000

Filtro de visualización: Enter a display filter ...

Copiar Guardar como... Cerrar

Clasificación de mensajes JSON-RPC

Tipo	Ejemplo
Sincronización	Request: {"jsonrpc":"2.0","id":1,"method":"initialize"} Response: {"jsonrpc":"2.0","id":1,"result":{...}}
Descubrimiento	Request: {"jsonrpc":"2.0","id":2,"method":"tools/list"} Response: {"jsonrpc":"2.0","id":2,"result":{...}}
Ejecución	Request: {"jsonrpc":"2.0","id":3,"method":"tools/call","params":{"name":"remote.ping","arguments":{}}} Response: {"jsonrpc":"2.0","id":3,"result":{"structuredContent":{"result":{"ok":true,"message":"pong"}}}}

A continuación se explica el flujo típico de una interacción entre el host y RemoteMCP, desglosado por capa del modelo OSI/TCP-IP:

Capa de enlace (Link)

La transmisión se realiza sobre un medio físico/lógico (Ethernet o Wi-Fi). Se observan tramas con direcciones MAC origen/destino y control de errores (FCS). La segmentación en tramas y el acceso al medio (CSMA/CD o CSMA/CA) son transparentes a las capas superiores.

Capa de red (IP)

Sobre la capa de enlace, IP enruta paquetes entre el host y los servidores de Cloudflare. Se aprecian direcciones IP origen/destino públicas, TTL decreciente en tránsito, y posible fragmentación si fuera necesario. La resolución de nombres (DNS) precede a la conexión (lookup de remote-mcp-demo.<subdominio>.workers.dev).

Capa de transporte (TCP/TLS)

El canal usa TCP (puerto 443). Se observa el three-way handshake (SYN, SYN/ACK, ACK), seguido por el establecimiento de la sesión TLS (ClientHello/ServerHello, intercambio de claves). Para descifrar el contenido en Wireshark se utilizó SSLKEYLOGFILE, permitiendo ver el tráfico WSS/HTTPS en claro.

Capa de aplicación (HTTP/WS + JSON-RPC + MCP)

Una vez establecido TLS, la aplicación utiliza HTTP/1.1 o HTTP/2 con upgrade a WebSocket (en WSS) o POST (en HTTP). El payload transporta mensajes JSON-RPC 2.0 con los métodos del protocolo MCP.

Clasificación de mensajes observados:

- - Sincronización: ``initialize`` (request) y su ``result`` (response).
- - Descubrimiento: ``tools/list`` (request) y su ``result`` (response) con metadatos de cada tool.
- - Ejecución: ``tools/call`` (request) con `{ name, arguments }` y su ``result`/`error`` (response).

Ejemplo de request/response (JSON-RPC):

```
{ "jsonrpc": "2.0", "id": 1, "method": "initialize" }  
{ "jsonrpc": "2.0", "id": 1, "result": { "serverInfo": { "name": "RemoteMCP-Demo", "version": "1.0.0" }, "capabilities": { "tools": {} } } }
```

Conclusiones y comentarios

El protocolo MCP permitió integrar de forma homogénea servidores locales y remotos, desacoplando la lógica de las herramientas del LLM/host. SiteLens demostró un caso no trivial de auditoría de HTML estático con reglas WCAG-lite, mientras que RemoteMCP evidenció la factibilidad de exponer herramientas por WSS/HTTP en la nube con baja latencia. La captura y análisis con Wireshark confirmó la secuencia de sincronización y ejecución en JSON-RPC sobre TLS, y ayudó a comprender la interacción de las capas. Entre los retos encontrados estuvieron el alineamiento de tipados (TypeScript vs. runtime de Workers), la normalización segura de rutas (whitelist de raíces) y el manejo explícito de schemas de entrada/salida para garantizar respuestas MCP válidas. En conjunto, el proyecto cumple los objetivos de comprender e implementar MCP en escenarios locales y remotos, integrándolo en un host basado en la API de OpenAI.

Referencias

1. Anthropic. (2025). Model Context Protocol Specification (2025-06-18).
<https://modelcontextprotocol.io/specification/2025-06-18>
2. JSON-RPC Working Group. (n.d.). JSON-RPC 2.0 Specification.
<https://www.jsonrpc.org/specification>

3. Model Context Protocol. (n.d.). Learn — Architecture.
<https://modelcontextprotocol.io/docs/learn/architecture>
4. Cloudflare. (n.d.). Cloudflare Workers Documentation.
<https://developers.cloudflare.com/workers/>
5. Universidad del Valle de Guatemala. (2025). Proyecto 1: Uso de un protocolo existente (Instrucciones de curso).

Apéndice A — Configuración del host (mcp_config.json)

```
{
  "servers": [
    {
      "name": "SQLScout",
      "transport": "stdio",
      "command": "python",
      "args": ["-B", "-m", "src.server_mcp"],
      "cwd": "C:/UVG/PROYECTO1/MCP_Local/SQL_MCP",
      "env": {}
    },
    {
      "name": "FS",
      "transport": "stdio",
      "command": "C:/Program Files/nodejs/npx.cmd",
      "args": ["-y", "@modelcontextprotocol/server-filesystem", "C:/UVG/PROYECTO1/MCP_Local"],
      "cwd": ".",
      "env": {}
    },
    {
      "name": "Git",
      "transport": "stdio",
      "command": "py",
      "args": ["-3.11", "-m", "mcp_server_git", "--repository", "C:/UVG/PROYECTO1/MCP_Local"],
      "cwd": "."
    },
    {
      "name": "SiteLens",
      "transport": "stdio",
      "command": "node",
      "args": ["C:/UVG/PROYECTO1/MCP_Local/sitelens/dist/server.js", "--
```



```
roots", "C:/UVG/PROYECTO1/MCP_Local/test/site"]
},
{
  "name": "anime-helper",
  "transport": "stdio",
  "command": "py",
  "args": ["-3.11", "-m", "anime_helper.server"],
  "cwd": ".",
  "env": {}
},
{
  "name": "RemoteMCP",
  "transport": "http",
  "url": "https://remote-mcp-demo.gabouvg.workers.dev/mcp"
}
]
}
```
