

Como Probar el Broker MQTT

Guia practica de como usar los scripts y comandos del kit contra broker.gilman.pro: que hace cada uno, como ejecutarlo y que esperar. Sin codigo fuente: solo instrucciones y comandos.

broker.gilman.pro · 2026-08-12

1. Antes de empezar

- **Python 3.10+** instalado en tu maquina.
- Instala la libreria MQTT (solo una vez):
`pip install paho-mqtt`
- **Mosquitto (CLI) es opcional** — el instalador para Windows esta incluido en la carpeta windows/. En Linux/macOS:
`sudo apt install mosquitto-clients`
- Descomprime el kit y trabaja desde la carpeta broker:
`cd kit-pruebas-mqtt/broker`

Todos los scripts funcionan con solo Python + paho-mqtt. Mosquitto solo es necesario si quieres usar mosquitto_pub / mosquitto_sub.

2. Los scripts del kit y como usarlos

Cada script hace una prueba concreta. Ejecutalo desde la carpeta **broker** y fíjate en la salida esperada:

Script	Que hace	Como ejecutarlo	Que esperar
publicador_password.py	Publica un mensaje por TCP (14769)	python3 publicador_password.py	"Conectado por TCP (broker.gilman.pro:14769)" + "Publicado en test/datos"
publicador_certificado.py	Publica por mTLS (14772) con los certificados incluidos	python3 publicador_certificado.py	"Conectado por mTLS (broker.gilman.pro:14772)"
suscriptor_certificado.py	Se suscribe por mTLS (14772) y muestra mensajes en vivo	python3 suscriptor_certificado.py	"Conectado por mTLS" + "Suscrito a 'test02/#' — queda escuchando (Ctrl+C para salir)"
test_wss.py	Prueba WSS (443) con PUB/SUB de ida y vuelta	python3 test_wss.py	"WSS OK: conectado a wss://broker.gilman.pro/mqtt" + "PUB/SUB OK"
validador_completo.py	Ejecuta las 16 pruebas: 5 protocolos + 10 cruzadas + intrusion	python3 validador_completo.py	"Total: 16/16 pruebas superadas"

Script	Que hace	Como ejecutarlo	Que esperar
mosquitto_wrapper.py	Emula mosquitto_pub/sub sin instalar nada (ver seccion 5)	python3 mosquitto_wrapper.py pub sub ...	Igual que los comandos CLI
sec_test_runner.py	Auditoria de seguridad web/API (en la carpeta security/)	cd ../security && python3 sec_test_runner.py	Resumen final con las pruebas superadas

3. Prueba rapida en 4 pasos

Paso 1 — TCP (14769): verifica la conexion basica con usuario y contrasena.

```
python3 publicador_password.py
```

Paso 2 — mTLS (14772): verifica la autentificacion mutua con certificados.

```
python3 publicador_certificado.py
```

Paso 3 — WSS (443): verifica el WebSocket seguro de produccion.

```
python3 test_wss.py
```

Paso 4 — Validacion completa: ejecuta las 16 pruebas de una sola vez.

```
python3 validador_completo.py
```

Si el paso 4 termina con "Total: 16/16 pruebas superadas", el broker esta operativo en los 5 protocolos y bloquea la intrusion mTLS.

4. Con mosquitto_pub / mosquitto_sub

Abre **dos terminales**. La Terminal 1 queda escuchando; la Terminal 2 publica el mensaje:

Terminal 1 — suscriptor:

```
mosquitto_sub -h broker.gilman.pro -p 14769 \
-u test01 -P "test01test01" -t "test/#" -v
```

Terminal 2 — publicador:

```
mosquitto_pub -h broker.gilman.pro -p 14769 \
-u test01 -P "test01test01" -t "test/hola" -m "Hola desde mi maquina" -q 1
```

Debes ver el mensaje aparecer en la Terminal 1.

4.1 mTLS desde CLI

```
mosquitto_pub -h broker.gilman.pro -p 14772 \
--cafile ca.crt --cert test02.crt --key test02.key \
-u test02 -P "test01test01" -t "test02/cli" -m "Hola mTLS"
```

4.2 WSS desde CLI

```
mosquitto_pub -L "wss://test01:test01test01@broker.gilman.pro:443/mqtt" \
-t "test/wss" -m "Hola WSS" --cafile cacert.pem
```

En WSS el certificado es de Let's Encrypt (nginx): usa el cacert.pem incluido en la carpeta broker/.

5. El wrapper (sin instalar Mosquitto)

mosquitto_wrapper.py emula mosquitto_pub / mosquitto_sub con paho-mqtt. Se usa igual que el CLI, con la palabra clave pub o sub:

```
python3 mosquitto_wrapper.py pub -h broker.gilman.pro -p 14769 \
-t "test/hola" -m "Hola desde mi maquina" -u test01 -P "test01test01"
```

```
python3 mosquitto_wrapper.py sub -h broker.gilman.pro -p 14772 \
-t "test/#" -u test02 -P "test01test01" \
--cert test02.crt --key test02.key --cafile ca.crt
```

6. Probar desde el navegador (Test-Broker)

Sin instalar nada: usa **Test-Broker** (test-broker.gilman.pro), el tester web de MQTT que se conecta a cualquier broker — incluido este.

- **1.** Solicita un usuario temporal en test-broker.gilman.pro/solicitar — llegan a tu correo (24 h).
- **2.** Inicia sesion y crea un cliente MQTT: host broker.gilman.pro, puerto 14769 (TCP), con el usuario recibido. Prueba tambien 14770 (WS), 14771 (TLS), 14772 (mTLS) y 443 (WSS).
- **3.** Publica y suscribete en tiempo real — 5 protocolos, MQTT v5 / v3.1.1.

El usuario temporal tambien sirve con los scripts: edita las constantes USERNAME / PASSWORD (o BROKER / PORT) de cada script y ejecuta normal.

7. Auditoria de seguridad (opcional)

Ejecuta el pentest automatizado para verificar TLS, API, headers, CORS y rate limiting del panel web:

```
cd ../security
python3 sec_test_runner.py
```

Al final muestra el resumen, por ejemplo: **19/21 pruebas superadas (90%)**.

8. Solucion de problemas

Problema	Causa	Solucion
Connection refused	Puerto incorrecto o firewall	Verificar puerto (14769-14772) y firewall
Not authorized	Credenciales incorrectas	Verificar username/password
Connection timeout	Sin conectividad	Verificar DNS y red
TLS: certificate verify failed	CA incorrecto	Usar --cafile con la CA correcta
WS: no se conecta	Falta libreria WebSocket	Usar transport='websockets' en Python
mTLS: handshake fallido	Certificado invalido	Verificar .crt y .key del dispositivo

Credenciales del kit: test01 / test01test01 (contrasena) y test02 / test01test01 (certificado). Son cuentas de prueba de este broker de demostracion; si no funcionan, solicita credenciales vigentes al administrador o pide un usuario temporal en Test-Broker (test-broker.gilman.pro/solicitar). No reutilices estas credenciales en produccion.