cailiujia / CVE Public

Code Issues 1 Pull requests Actions Projects Security and quality Insights

... 1 Branch 0 Tags Go to file Go to file Code

cailiujia Add files via upload 2edb031 · 3 weeks ago

demo.mp4	Add files via upload	3 weeks ago
mqtt_user_vs_clientid.py	Add files via upload	3 weeks ago
readme.md	Update readme.md	3 weeks ago

README

EMQX Enterprise Cross-User Client ID Conflict – Denial of Service (DoS)

📌 Vulnerability Overview

EMQX Enterprise versions 6.1.0 and earlier contain an access control flaw caused by improper session management. **An authenticated attacker** can use any valid MQTT account to connect to the broker using a victim's Client ID, forcing the victim's device to be disconnected. This results in a denial-of-service condition.

The root cause is that EMQX treats the Client ID as the sole session identifier without binding it to the authenticated username. In multi-tenant or multi-user environments, this allows one user to interfere with another user's connections.

🎯 Affected Versions

- **EMQX Enterprise** 6.1.0 (earlier versions may also be affected)
- Test tools: MQTTX 1.13.0, Mosquitto Client 1.6.0

⚠️ Impact

- The attacker does not need the victim's password—only a valid account and knowledge of the target Client ID.
- In IoT or connected vehicle scenarios, critical devices can be taken offline, causing service disruption.
- Attackers can enumerate or guess common Client ID patterns (e.g., serial numbers, product IDs) to affect many devices at once.

🔁 Reproduction Steps

1. **User A** (legitimate) connects to the EMQX broker with Client ID `Device001` using their own credentials. The connection succeeds.
2. **User B** (attacker) connects to the same broker with the **same Client ID** `Device001` but using their own credentials (no knowledge of User A's password is required).
3. Result: After User B connects, User A's connection is immediately terminated. The broker log shows the old connection being replaced by the new one.

Proof-of-Concept Script

The repository includes the Python script `mqtt_user_vs_clientid.py` to simulate multiple client connections and quickly verify the vulnerability.

Dependencies

```
pip install paho-mqtt
```

Configuration

Edit the script and replace the following variables with your broker details:

```
USERNAME = "your_username"
PASSWORD = "your_password"
BROKER = "your_broker_ip"
PORT = 1883
CLIENT_IDS = ["Client1", "Client2", "Client3"]   # Client IDs to test
```

Running the Script

To reproduce the vulnerability step by step, follow this workflow:

Step 1 — Start the Vulnerable EMQX Broker

```
# Pull the affected version (6.1.0)
docker pull emqx/emqx-enterprise:6.1.0

# Run it in the background
docker run -d --name emqx-test -p 1883:1883 -p 18083:18083 emqx/emqx-enterprise:6.1.0
```

Wait ~20 seconds for EMQX to initialize. You can check status with:

```
docker logs emqx-test | grep "EMQX .* is running"
```

> **Default dashboard:** http://localhost:18083 — credentials: `admin` / `public`

Step 2 — Install and Use MQTTX Desktop (Simulating the Victim)

Download and install **MQTTX Desktop v1.13.0** (or a compatible version) from:
https://www.emqx.com/en/downloads/mqttx

After installation, launch MQTTX and set up **three connections** for the legitimate user `abc`:

1. Click **+ New Connection**.
2. Fill in the fields as shown below for each connection:

Field	Connection 1	Connection 2	Connection 3
Name	abc - client1	abc - client2	abc - client3
Client ID	Client1	Client2	Client3
Host	localhost	localhost	localhost
Port	1883	1883	1883
Username	abc	abc	abc
Password	password123	password123	password123

3. Click **Connect** for each connection to establish the session.

> **Keep all three connections active in MQTTX.** These represent the victim's legitimate sessions.

Step 3 — Launch the Attack (Using a Different User)

Edit `mqtt_user_vs_clientid.py` with the attacker's credentials, targeting the **same Client IDs** used by the victim:

```
USERNAME = "attacker"      # Must be different from 'abc'
PASSWORD = "evilpass"
BROKER = "localhost"
PORT = 1883
CLIENT_IDS = ["Client1", "Client2", "Client3"]  # Same Client IDs as the victim above
```

Then run the attack script:

```
python mqtt_user_vs_clientid.py
```

> **Expected Result:** As soon as the script connects, the corresponding sessions in MQTTX (`abc - client1`, `abc - client2`, `abc - client3`) will **automatically disconnect** — even though they were authenticated with valid credentials. This confirms the cross-user DoS via Client ID collision.

The script creates one connection per Client ID in the list. To observe the conflict in a more controlled manner, you can also run the script **twice in separate terminals** using two different user accounts (modify `USERNAME` / `PASSWORD` accordingly).

📹 Demo Video

The file `demo.mp4` in this repository shows a full demonstration of the vulnerability.

🛡️ Recommended Fixes

1. Enforce a composite unique constraint combining Client ID with username (or tenant ID) so that different users can use the same Client ID without interfering.
2. Add a broker configuration option to bind Client IDs to specific users, preventing cross-user session takeover.

Releases

No releases published

Packages

No packages published

Contributors 1

cailiujia

Languages

- **Python** 100.0%