# Hardcoded Authentication Key Allows Unauthorized Bluetooth Unlocking of Kuiyue Shared Type 2 Devices

## Vulnerability Summary

Kuiyue Shared uses a fixed AES-128 key embedded in its WeChat mini-program client to authenticate type 2 ECU Bluetooth unlocking sessions. A nearby party who obtains this client-side constant can generate a valid response to the current device challenge and issue the unlock command. The authentication exchange does not bind device control to the current user or a server-issued authorization for the requested unlock.

We confirmed physical unlocking using a standalone BLE implementation. This demonstrates unauthorized control of the type 2 ECU unlock function.

Supplier: 杭州几何平方科技有限公司, described in English as Hangzhou Jihe Pingfang Technology Co., Ltd. Product: Kuiyue Shared (奎越共享), type 2 ECU BLE unlocking. Client identifier: WeChat AppID wx740a7a2f10681592. Report date: 4 November 2026.

## Normal Unlocking and Protocol Flow

The mini-program uses BLE for recovery unlocking in an abnormal-order workflow. It obtains the target device information, authenticates locally, performs the unlock and reports the result to the backend after device-side completion.

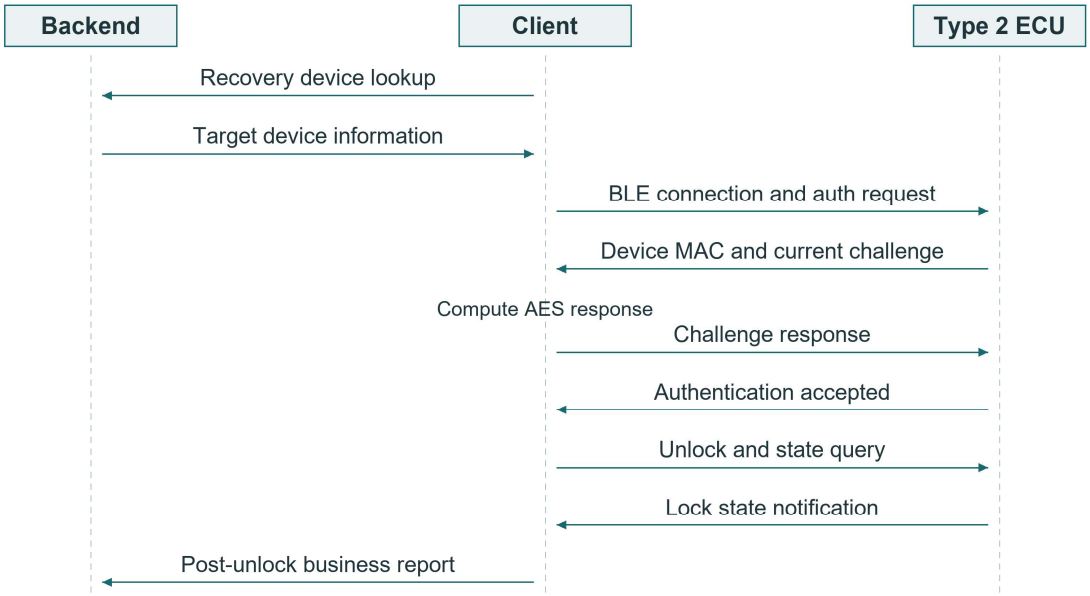



Figure 1 Type 2 BLE recovery flow with backend reporting after device unlocking

1. The client obtains the target device information and connects to the type 2 ECU through BLE service FFF0 and characteristic FFF6, then subscribes to notifications.
2. The device returns a 6-byte authentication MAC and a 4-byte challenge. The connection address and the authentication MAC are distinct fields.
3. The client concatenates the authentication MAC, challenge and six zero bytes into one 16-byte block and encrypts it with the embedded AES-128 key in ECB mode without additional padding.
4. The client submits a 21-byte response containing the ciphertext and an XOR checksum, using the existing 20-byte plus 1-byte write sequence. A 4A / 00 response is treated as authentication success.
5. After authentication, the client sends the fixed unlock command and queries the lock state. The normal application then reports successful BLE unlocking to the backend.

### Parameter Source Analysis

| Parameter | Source | Function |
|---|---|---|
| Connection address | Obtained from the target device's BLE connection information. | Selects the BLE device to connect to; does not establish business authorization. |
| Authentication MAC | Six bytes returned by the ECU in its authentication response. | Forms part of the plaintext used to compute the challenge response. |
| Challenge | Four bytes returned by the ECU for the current authentication exchange. | Provides a fresh input; does not compensate for an exposed authentication key. |
| Authentication key | A fixed 16-byte AES key embedded in the type 2 client implementation. | Allows an independent client to compute accepted responses. |
| Authentication response | AES-ECB encryption of the MAC, challenge and zero padding, plus frame checksum. | Establishes the device-side authenticated state. |
| Unlock command | A fixed control message used after authentication. | Requests physical unlocking without carrying a user or order authorization. |
| Lock state | A device status notification; value 01 is treated as unlocked by the client. | Reports the device's lock state. |

## Attack Flow

1. Obtain the fixed authentication key and protocol structure from the distributed type 2 client.
2. Within BLE range, connect directly to the target ECU and request its current authentication data through the specified GATT interface.
3. Use the MAC and current challenge supplied by that ECU to calculate a valid authentication response with the exposed key.
4. After the ECU accepts the response, send the fixed unlock command and request a new status notification.

## Core Security Issue

The device treats possession of a client-embedded constant as sufficient proof of authorization. The challenge-response mechanism can verify that the peer knows that constant, but it does not establish that the peer is the current authorized user or holds permission for this device and this unlock operation.

Fresh challenges do not resolve this weakness. Once the fixed key is available, an independent client can calculate a new valid response for each challenge. The issue therefore does not depend on breaking AES or replaying an old authentication transcript.

The authentication plaintext contains only device-supplied values and zero padding. It carries no user identity, order identifier, validity period or server-issued authorization. Backend reporting occurs after the device operation, so that reporting step cannot provide the missing device-side authorization check before unlocking.

## Experimental Validation and Evidence

### Protocol Review

The existing standalone implementation follows the type 2 connection, challenge-response, unlock and status-query sequence. Its relevant authentication inputs are the fixed client key and the MAC and challenge returned by the device. It performs no login, order-creation or post-unlock reporting request to the business backend.

### Physical Unlocking Result

We confirmed successful physical unlocking using the standalone BLE workflow.

### Authentication and Lock Status

The program treats command 4A with status 00 as authentication success and a newly received complete status frame with state 01 as unlocked.

## Impact and Affected Product

An attacker within BLE communication range can control the device's physical unlock function using the exposed client key and protocol. Exploitation requires a connection to the relevant GATT interface.

The affected product is Kuiyue Shared type 2 ECU BLE unlocking, associated with WeChat AppID wx740a7a2f10681592.

## Recommended Remediation

Remove the fixed authentication key from the distributed client and require the device to verify trusted authorization for each unlock. Bind that authorization to the target device, current session, requested operation, valid order, expiration time and replay-protection data. Protect device-specific long-term secrets from disclosure to ordinary clients.

Coordinate device firmware and service changes, reject unlock requests that lack valid business authorization and retire the legacy authentication path that accepts responses based on the exposed constant. Operators can evaluate restricting the affected BLE control path at the device level while a correction is prepared.

### Weakness Classification

CWE-321  Use of Hard-coded Cryptographic Key. The immediate root cause is the embedded fixed key used for device authentication; unauthorized unlocking is the demonstrated consequence.