1 Vulnerability analysis
2 The tinycve application performs two queries on a vulnerability database. The first query is on the basis of a search string, and it is uses a regular expression. The second query is on the basis of a the client's "platform", which the application seems to automagically guess. It turns out that tinycve guesses it from the "platform" field of the "User-Agent" HTTP request header. Such a field is the one between the first '(' character and the successive ';' character, for example "X11" in:
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/51.0.2704.103 Safari/537.36
4 The first query is filtered through a mysql_real_escape_string() function, but the second query is vulnerable to SQL injection. Indeed, the platform string is concatenated unsafely to a SQL query, allowing for SQL injection into string literal constant. To be really sure, we can test the vulnerability (with Burp Repeater) by sending a quote character:
5 User-Agent: Mozilla/5.0 ('; Linux x86_64; rv:78.0) Gecko/20100101 Firefox/78.0
6 We can observe that the response contains an "Invalid query" error message.
7