# ransom6 [50]

Task description: ./description.txt

TL;DR : blind SQL injection (or error based injection, but I used blind during the contest, will describe error based too)

Resources: ./ransom6.php

We are presented with a website which verifies a btc wallet handle and checks if the corresponding sum is paid. I immediately thought of an SQL injection, and inputting ‘ into the form does the job, receiving an **Fatal error: Call to a member function fetch_assoc().** After a few tries I can see that the script will return 2 texts depending of the outcome of the query (true/false) or an error, so I am thinking of blind SQLi.

I wrote a quick PHP script to “abuse” the bug, and bruteforce the database structure and some necessary rows. (see ransom6.php, $query[1-6]). I found out that there is a table victim with 3 columns, **btc-wallet**, **agent** and **status**. Thinking that status refers to the state of the ransom being paid or not, I tried off my head the following: ‘ or status=1# and it worked indeed, getting the flag format: **The flag is in this format CSC2016{sha256(`admin_user-agent`)};** . So I need the useragent of admin, and since the website doesn’t allow users to add wallets in the database, I can assume that the only useragent present there will be the admin’s one. Said and done, I extracted the first useragent: “**Mozilla/10.0 Gecko/20100101 Firefox/1337.0**”, sha256’d it and submitted the flag as the correct one.

Another interesting solution (and quicker, easier one) is the error-based solution. Basically, we trick SQL into showing an error about our query after processing a subquery, ABOUT that subquery, which will show us it’s result. For example, using: **' or 1 group by concat_ws(0x3a,(version()),floor(rand(0)*2)) having min(0) or 1#** , database() will be processed, then a duplicate entry error will be shown, Duplicate entry **'5.6.30-0ubuntu0.15.10.1**:1' for key 'group_key' and there we go. From here, just replace **version()** with **(select agent from victim limit 1)**, and the useragent will appear.

Final flag: **CSC2016{734518cf2bb140ddccd2300e252cabbc5ddc59156624700f53b42a2cca708f3b}**