CSE 5472: SQL Injection Lab

Objective

Learn about vulnerabilities stemming from control/data plane confusion via blind SQL injections.

Deliverable

1. Completed question sheet.
2. Report describing how each question was solved with supporting screenshots of executed terminal commands.

Environment

Please use a Linux environment (e.g., Debian, Ubuntu). A virtual machine or `stdlinux` is fine too.

Provided Materials

- `question-sheet.txt`: A text file for writing your solutions to the tasks.
- `database.db`: A SQLite database with usernames and hashed passwords.
- `authenticate.py`: A simple Python script that'll act as a surrogate for a website's login page.

Recommended Tools

- hashcat or John the Ripper

Note: `hashcat` **will not** work on `stdlinux`. If you are using `stdlinux`, please use John the Ripper and see the hints at the end of this document for compilation instructions.

Note: In some Linux environments, `hashcat` can be installed using APT: `sudo apt install hashcat`.

Tasks

1. Complete the tasks in `question-sheet.txt`.

Note: Solutions that modify `authenticate.py` or directly read `database.db` will recieve no credit!

Note: For Task 4, you **must** use blind SQL injection. Solutions that rely on `UNION` or `JOIN` and require knowing the names of table columns in advance will not recieve credit.

Note: You are not allowed to use any pentesting tools to complete the SQL injection tasks. You *may* use Hashcat, John the Ripper, or a similar tool to crack hashes.

Grading

- Question Sheet (See `question-sheet.txt` for grading breakdown)

Hints

- `authenticate.py` returns one of two messages (and exit codes) depending on whether the SQLite query it performed returned any matches or not.
- Two useful SQLite commands that can help you complete this lab are `SUBSTR` and `LENGTH`.

How to Compile John the Ripper on `stdlinux`:

```
git clone https://github.com/openwall/john
cd john/src/
```

```
./configure --without-openssl
make -j4
```

You should now have a compiled binary at `../run/john`.