0. 建议在 Linux 环境下完成实验（推荐顺序是 Fedora>Ubuntu>其他 Linux 发行版>Kali or BT)。windows 下如果能搞定配置也是可以接受的。

1. 搭建 mysql 数据库，建立数据库 test，数据表 student，包含 id、name、score 三列。

2. 搭建应用的运行环境，如 nginx+php-fpm、tomcat+java 等等。

3. 编写带有 sql 注入漏洞的接口程序，包含：
   - 根据输入的参数值，拼接 SQL 查询语句并执行，将查询结果展示。如根据输入的学号展示姓名和分数。
   - 根据输入的参数值，拼接 SQL 查询语句并执行，展示查询结果是否为空。如输入学号，展示是否有该学生存在。
   - 根据输入的参数值，拼接 SQL 查询语句并执行，将查询结果是否为空展示在两段随机内容之间。
   - 根据输入的参数值，拼接 SQL 查询语句并执行，展示查询结果的条件表达式结果，并将结果展示在两段随机内容之间。如入学号，展示该学生分数是否大于 60。
   - 根据输入的参数值，拼接 SQL 查询语句并执行，但展示一个固定的结果。如如输入学号，查询是否有学生存在，然后输出固定内容。
   - 据输入的参数值，拼接 SQL 语句并执行，更新数据库。如输入学号和分数，将对应学生的分数更新。

4. 针对上述各个应用接口，手工修改请求参数，尝试各种 SQL 注入的攻击向量，和正常访问的对照组一起，观察结果并记录。

5. 针对上述各个应用接口， 用 sqlmap 尝试各种注入方式，并用 wireshark 或 ZAP 抓包，记录每次的目标、SQL 命令行、结果（包括出结果的过程、和最终的输出）、和抓包文件。

6. 分析抓包文件，了解攻击向量，结合 mysql 查询日志和 mysql 控制台，体会各种注入技术的原理。并回到 4 步骤中手动尝试。

6.5 选做，在步骤 3 中第一个接口的基础上，尝试用不同的方法来避免 SQL 注入，再使用 sqlmap 尝试看是否有效，能否绕过。

7. 写实验报告，看如何用一篇文档以及附件来讲明白上述实验的动机、步骤、结果、和分析结论 。

如果搞不定，0-5 的步骤可以找搞定了的同学求助，注明从谁那得到的哪种程度的帮助即可，不影响成绩评定。

上面实验设计如果有不好、不对或者可以更好的地方，也可以自行改进。