**西安邮电大学**

**网络空间安全学院**

**实验报告**

**（2019/ 2020学年第1学期）**

实验名称： 信息安全专业课程设计A

学生姓名： XXXXXX c

专 业： 安全XXXX班

学 号： 8位学号（班内序号）

指导教师： XXX

**西安邮电大学网络空间安全学院XXXX（实验课程名）**

**考核表**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **学生姓名** |  | **班级/学号** | | **/** | |
| **承担任务实验室（单位）** | **信息安全与信息对抗实验教学中心** | **所在部门** | | **网安院XXX** | |
| **实施时间** | **年 月 日 — 年 月 日** | | | | |
| **实验内容** |  | | | | |
| **指导教师（师傅）姓名** |  | **职务或职称** | |  | |
| **指导教师（师傅）**  **对学生的评价** | **评价点** | **评价内容** | | | **评价得分** |
| **平时成绩**  **（20分）** | **课堂出勤、课堂纪律及学习态度**  **遵守实验室的规章制度等情况** | | |  |
| **实验能力**  **（30分）** | **文件检索或软硬件设计能力**  **分析、解决实际问题能力** | | |  |
| **实验验收**  **（20分）** | **实验最终完成质量情况**  **实验演示与答辩情况** | | |  |
| **实验报告**  **（30分）** | **语言及文字表达能力**  **实验报告结构及书写格式规范** | | |  |
| **总评成绩** |  | | **五级制** |  | |
| **指导教师（师傅）评语** | **指导教师（师傅）签字**  **年 月 日** | | | | |

1. **实验目的和要求**

**（实验报告正文可根据具体实验适当调整标题与格式。要求每次实验结束后需提交一次实验报告，全部实验结束后各次实验报告统一装订，附考核表一份，手写还是打印由课程教师确定）**

**正文要求小四号宋体，1.5倍行距。**

页面设置：上－2.5厘米，下－2厘米，左－2.5厘米，右－2厘米，装订线－0厘米，页眉－2厘米，页角－2厘米。

一级标题（章）：用1、2、3、4、5、6……的序号，小三号的加黑的黑体字，居中。一级标题（章）之间另起一页。

二级标题（节）：用1.1、1.2、1,3、1.4……的序号，四号的加黑的黑体字，居中。和一级标题之间空一行。

三级标题：用1.1.1、1.1.2、1.1.3、1.1.4……的序号，小四号的加黑的黑体字，靠左。和二级标题之间空一行。

四级标题：用1.1.1.1、1.1.1.2、1.1.1.3、1.1.1.4……的序号，小四号的宋体字，靠左。

作为章节和段落的序号划分至多到四级。

页码：用阿拉伯数字连续编页，用小五号Times New Roman字号，居中，数字两侧不加任何修饰。

**插图：**插图包括图解、示意图、构造图、曲线图、框图、流程图、布置图、地图、照片、图版等。插图注明项有图号、图题、图例。图号编码用章序号。如“图2.1”表示第2章第1图。图号与图题文字留一字空格置于无色文本框内，放在图的正下方居中，最好和图组合在一起，图题用5号或小5号字，字体用宋体。图中标注符号文字字号不大于图题的字号。

为确保打印足够清晰的图，还需注意以下几点：若是扫描图，则要保证有大于300dpi的分辨率；若是编程软件绘制的图，则应先画出足够大的图，再通过拷贝屏幕的方式，其它根据具体情况而定。

**表：**表的一般格式是数据依序竖排，内容和项目由左至右横读，通版排版。表号也用章序号编号码，如：表2.1是第2章中的第1表。表应有表题，与表号之间空1字，置于无色文本框内，放在表的上方居中，用5号或小5号宋体。表中的内容和项目字号不大于图题的字号。

**公式：**公式包括数学、物理和信息学等。正文中引用的公式、算式或方程式等可以按章序号用阿拉伯数字编号（式号），如：式（2.1）表示第2章第1式，公式一般单行居中排版与上下文分开，式号与公式同行，居右排版。

1. **实验原理**

**2.1 通信：通过socket实现服务器与客户端的远程通信，通过多线程可同时服务多个客户端**

**2.2. 身份验证：将允许的客户端口令（账号）保存到文件中，等有客户端进行连接时先进行验证，若没有改账号，则直接拒绝**

**2.3 日志记录：将用户信息写到日志文件中，同时通过多线程启动一个专门读取日志的线程，从而在保证服务线程正常运行的情况下，在控制台打印日志信息**

1. **实验内容及步骤**

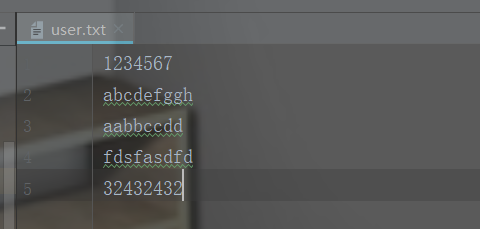
**3.1 服务端**

**ServerMain：负责启动服务端**

1. **import** java.io.IOException;
3. **public** **class** ServerMain {
4. **public** **static** **void** main(String[] args) {
5. **try** {
6. **new** Server().start();
7. } **catch** (IOException e) {
8. e.printStackTrace();
9. }
10. }
11. }

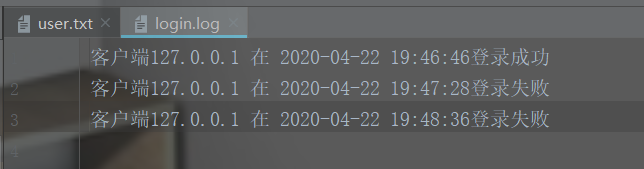
**ServerThread：服务端接收客户请求线程**

1. **接收客户端请求，并读取客户端的口令信息**
2. **对读取的客户端口令进行验证，即与user.txt的允许口令一一比对，返回true 或 false**



1. **将登陆信息写入日志login.log，这里是追加，并不会覆盖已有的日志信息**

**如图，login.log中已经有了部分日志信息**



1. **import** java.io.\*;
2. **import** java.net.Socket;
3. **import** java.text.SimpleDateFormat;
4. **import** java.util.Date;
5. **import** java.util.logging.Logger;
7. **public** **class** ServerThread **extends** Thread {
9. **private** Socket conn;
11. **private** **static** **final** String name = ServerThread.**class**.getName();
12. **private** **static** **final** Logger log = Logger.getLogger(name);
14. **public** ServerThread(Socket conn) {
15. **this**.conn = conn;
16. log.info(conn.getInetAddress().getHostAddress() + " 尝试登录ing");
17. }
19. @Override
20. **public** **void** run() {
21. **try** {
22. writeLog(check(read()));
23. } **catch** (IOException e) {
24. e.printStackTrace();
25. }
26. }
28. **public** String read() **throws** IOException {
29. InputStream inputStream = conn.getInputStream();
30. BufferedReader bufferedReader = **new** BufferedReader(**new** InputStreamReader(inputStream));
31. String token = bufferedReader.readLine();
33. bufferedReader.close();
34. inputStream.close();
36. **return** token;
37. }
39. **public** Boolean check(String token) **throws** IOException {
40. File file = **new** File("server03/src/user.txt");
41. FileReader reader = **new** FileReader(file);
42. BufferedReader bufferedReader = **new** BufferedReader(reader);
44. String info;
45. **while** ((info = bufferedReader.readLine()) != **null**) {
46. **if** (info.equals(token)) {
47. log.info(conn.getInetAddress().getHostAddress() + " 登录成功");
48. **return** **true**;
49. }
50. }
51. log.info(conn.getInetAddress().getHostAddress() + " 登录失败");
52. **return** **false**;
53. }
55. **public** **void** writeLog(Boolean success) **throws** IOException {
56. File file = **new** File("server03/src/login.log");
57. FileWriter fileWriter = **new** FileWriter(file, **true**);
58. PrintWriter printWriter = **new** PrintWriter(fileWriter);
60. SimpleDateFormat simpleDateFormat = **new** SimpleDateFormat("yyyy-MM-dd HH:mm:ss");
61. String now = simpleDateFormat.format(**new** Date());
62. String addr = conn.getInetAddress().getHostAddress();
63. String msg = success ? "登录成功" : "登录失败";
64. printWriter.println("客户端" + addr + " 在 " + now + msg);
66. printWriter.close();
67. fileWriter.close();
68. }
70. }

**QueryThead：服务端查询日志的线程，只要在控制台输入Q就会打印日志信息；**

**且不影响ServerThread**

1. **import** java.io.\*;
2. **import** java.util.Scanner;
3. **import** java.util.logging.Logger;
5. **public** **class** QueryThread **extends** Thread {
7. **private** **static** **final** String name = ServerThread.**class**.getName();
8. **private** **static** **final** Logger log = Logger.getLogger(name);
10. @Override
11. **public** **void** run() {
12. **try** {
13. log.info("日志已经开始写入login.log，查询请按 Q ！！！");
14. readLog();
15. } **catch** (IOException e) {
16. e.printStackTrace();
17. }
18. }
20. **public** **void** readLog() **throws** IOException {
21. File file = **new** File("server03/src/login.log");
22. FileReader fileReader = **new** FileReader(file);
23. BufferedReader bufferedReader = **new** BufferedReader(fileReader);
24. Scanner in = **new** Scanner(System.in);
25. **while** ((in.next().equalsIgnoreCase("q"))) {
26. //            for(int i = 0; i < 15; i++) System.out.println();
27. System.out.println("历史日志信息如下（login.log）");
28. String log;
29. **while** ((log = bufferedReader.readLine()) != **null**) {
30. System.out.println(log);
31. }
32. }
33. }
34. }

**Server：负责启动一个QueryThread，与多个ServerThead**

1. **import** java.io.\*;
2. **import** java.net.ServerSocket;
3. **import** java.net.Socket;
4. **import** java.util.logging.Logger;
6. **public** **class** Server {
8. **private** **static** String name = Server.**class**.getName();
9. **private** **static** Logger log = Logger.getLogger(name);

12. **public** **void** start() **throws** IOException {
13. ServerSocket server = **new** ServerSocket(8080);
14. log.info("服务器在8080端口启动");
15. **new** QueryThread().start();

18. **while** (**true**) {
19. Socket conn = server.accept();
20. **new** ServerThread(conn).start();
21. }
22. }
23. }

**3.2客户端**

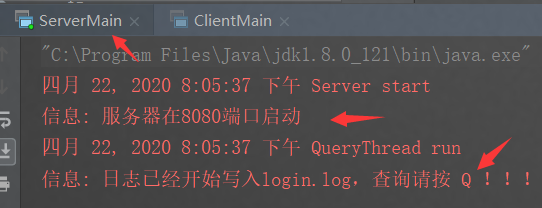
**ClientMain：启动客户端**

1. **import** java.io.IOException;
3. **public** **class** ClientMain {
5. **public** **static** **void** main(String[] args) {
6. **try** {
7. **new** Client().start();
8. } **catch** (IOException e) {
9. e.printStackTrace();
10. }
11. }
12. }

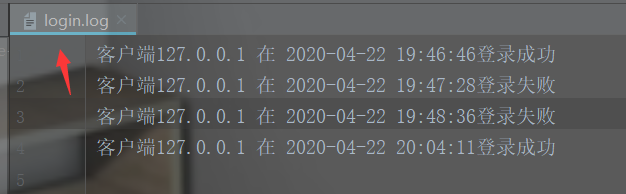
**Client：读取用户输入并将口令发送到服务端**

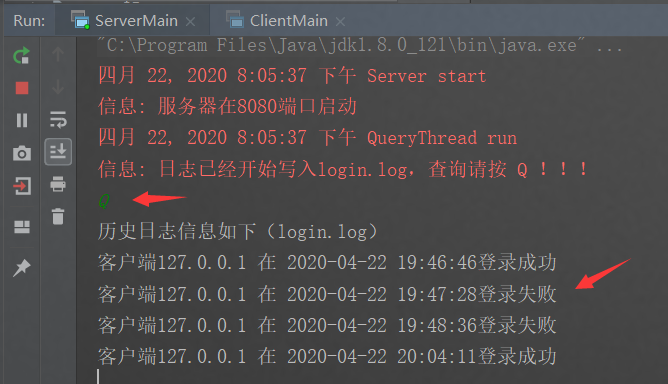
1. **import** java.io.IOException;
2. **import** java.io.OutputStream;
3. **import** java.io.PrintWriter;
4. **import** java.net.Socket;
5. **import** java.util.HashMap;
6. **import** java.util.Map;
7. **import** java.util.Scanner;
8. **import** java.util.logging.Logger;
10. **public** **class** Client {
12. **private** **static** String name = Client.**class**.getName();
13. **private** **static** Logger log = Logger.getLogger(name);
15. **public** **void** start() **throws** IOException {
16. Map<String, Object> input = input();
17. Socket client = **new** Socket((String)input.get("ip"), (Integer)input.get("port"));
18. log.info("客户端启动。。。");
19. OutputStream outputStream = client.getOutputStream();
20. PrintWriter printWriter = **new** PrintWriter(outputStream);
21. printWriter.println((String)input.get("token"));
22. printWriter.flush();
23. log.info("账号口令已发送");
24. }
26. **public** Map<String,Object> input() {
27. HashMap<String, Object> map = **new** HashMap<>();
29. Scanner in = **new** Scanner(System.in);
30. System.out.print("服务器IP：");
31. map.put("ip",in.next());
32. System.out.print("服务器端口：");
33. map.put("port",in.nextInt());
34. System.out.print("账号：");
35. map.put("token",in.next());
37. **return** map;
38. }
39. }
40. **实验结果分析**

**1.启动服务端**

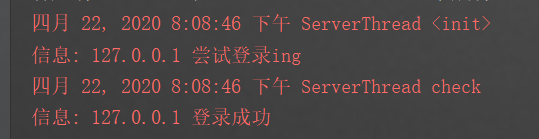
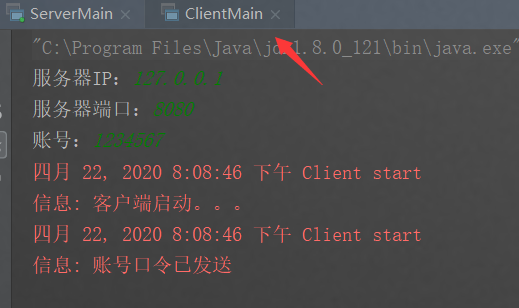


**2.查看现有的日志信息**

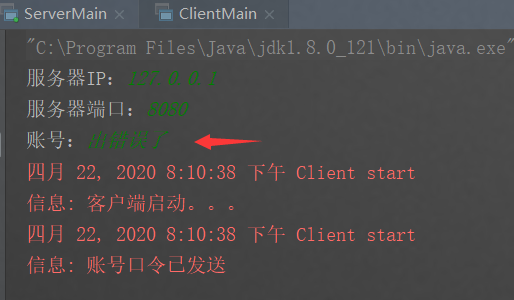




**3.启动客户端，读取输入发送口令；并观察服务端**

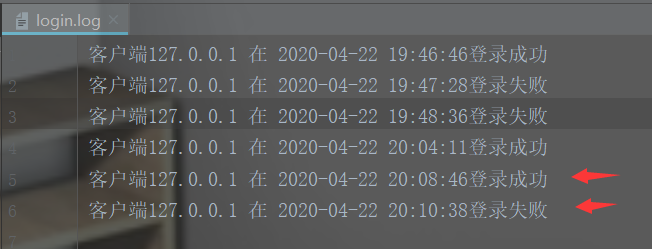


**这是口令1234567在user.txt中，若一个不存在的口令登录时**





**4.查看此时的日志信息**





**五．实验总结及心得体会**