**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 进行服务器与客户端的通信，并且通过多线程，服务器可以同时服务多个客户端**

**2.2 登录校验：客户端登录要发送口令，在服务端存有user.txt保存了所有允许的口令；当有客户端要登录时，与user.txt逐一对比，并非客户端返回状态码**

**2.3 数据嗅探：在服务端可以输入敏感词，并与所有客户端的消息进行比对，若包含则打印**

1. **实验内容及步骤**

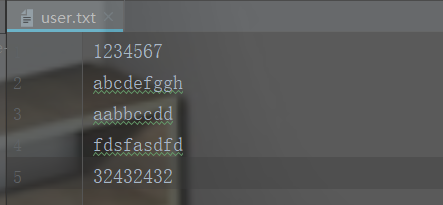
**3.1 Server**

**ServerMain：用来启动服务端**

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

**ServerThread：连接客户端线程**

1. **登录校验，将用户的口令与user.txt进行比对，若成功则向客户端返回1，若失败则向客户端返回0并断开连接**



1. **敏感词校验，不断接收客户端的消息，直到客户端退出（quit）；将消息与敏感词比对，若包含敏感词则将消息打印**
2. **import** java.io.\*;
3. **import** java.net.Socket;
4. **import** java.util.logging.Logger;
6. **public** **class** ServerThread **extends** Thread {
8. **private** Socket conn;
9. **private** String keyWord;
11. **private** **static** **final** String name = ServerThread.**class**.getName();
12. **private** **static** **final** Logger log = Logger.getLogger(name);
14. **public** ServerThread(Socket conn,String keyWord) {
15. **this**.conn = conn;
16. **this**.keyWord = keyWord;
17. }
19. @Override
20. **public** **void** run() {
21. **try** {
22. work();
23. } **catch** (IOException e) {
24. e.printStackTrace();
25. }
26. }
28. **public** **void** work() **throws** IOException {
29. InputStream inputStream = conn.getInputStream();
30. BufferedReader reader = **new** BufferedReader(**new** InputStreamReader(inputStream));
31. **if** (!check(reader.readLine().trim())) **return**;
33. **while** (**true**) {
34. String msg;
35. **if** ((msg = reader.readLine()).contains(keyWord)) {
36. System.out.println(conn.getInetAddress().getHostAddress() + " - 敏感：" + msg);
37. } **else** **if** (msg.equalsIgnoreCase("quit")) {
38. log.info(conn.getInetAddress().getHostAddress() + " 连接断开。。。");
39. **break**;
40. }
41. }
43. reader.close();
44. inputStream.close();
45. }
47. **public** Boolean check(String token) **throws** IOException {
48. File file = **new** File("server04/src/user.txt");
49. FileReader reader = **new** FileReader(file);
50. BufferedReader bufferedReader = **new** BufferedReader(reader);
51. OutputStream outputStream = conn.getOutputStream();
52. PrintWriter printWriter = **new** PrintWriter(outputStream);
54. String info;
55. **while** ((info = bufferedReader.readLine()) != **null**) {
56. **if** (info.equals(token)) {
57. log.info(conn.getInetAddress().getHostAddress() + " 登录成功");
58. printWriter.println(1);
59. printWriter.flush();
60. **return** **true**;
61. }
62. }
63. log.info(conn.getInetAddress().getHostAddress() + " 登录失败");
64. printWriter.println(0);
65. printWriter.flush();
67. printWriter.close();
68. outputStream.close();
69. bufferedReader.close();
70. reader.close();
71. **return** **false**;
72. }


76. }

**Server：启动连接客户端的线程们**

1. **import** java.io.IOException;
2. **import** java.net.ServerSocket;
3. **import** java.net.Socket;
4. **import** java.util.Scanner;
5. **import** java.util.concurrent.TimeUnit;
6. **import** java.util.logging.Logger;
8. **public** **class** Server {
10. **private** **static** String name = Server.**class**.getName();
11. **private** **static** Logger log = Logger.getLogger(name);
13. **public** **void** start() **throws** IOException, InterruptedException {
14. ServerSocket server = **new** ServerSocket(8080);
15. log.info("服务器在8080端口启动");
16. Scanner in = **new** Scanner(System.in);
17. TimeUnit.SECONDS.sleep(1);
18. System.out.print("请输入要过滤的敏感词：");
19. String keyWord = in.next();
20. System.out.println("保存成功！");
22. **while** (**true**) {
23. Socket conn = server.accept();
24. **new** ServerThread(conn,keyWord).start();
25. }
27. }
28. }

**3.2 Client**

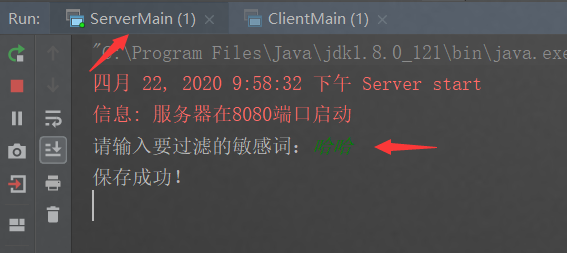
**ClientMain:启动客户端**

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. }

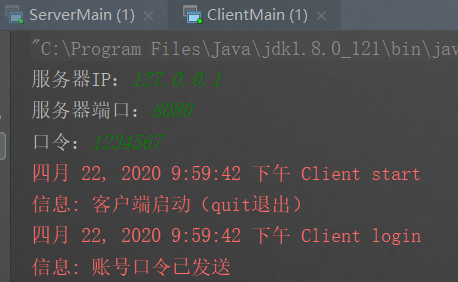
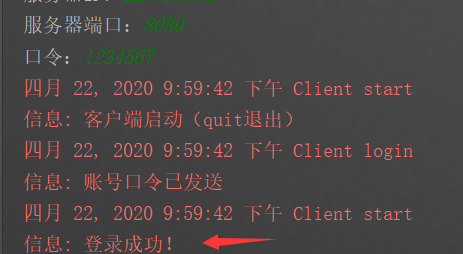
**Client**

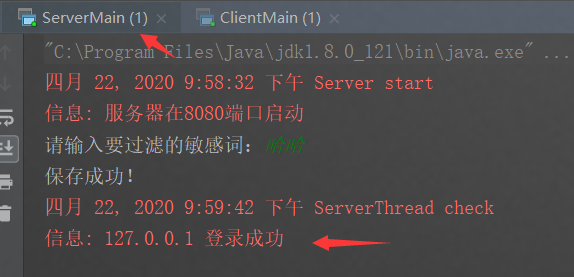
1. **读取用户输入**
2. **连接服务器，尝试登录**
3. **若登录成功，则对服务端发送消息（quit退出）**
4. **import** java.io.\*;
5. **import** java.net.Socket;
6. **import** java.util.HashMap;
7. **import** java.util.Map;
8. **import** java.util.Scanner;
9. **import** java.util.logging.Logger;
11. **public** **class** Client  {
13. **private** Socket client = **null**;
15. **private** **static** String name = Client.**class**.getName();
16. **private** **static** Logger log = Logger.getLogger(name);
18. **public** **void** start() **throws** IOException {
19. Map<String, Object> input = input();
20. client = **new** Socket((String)input.get("ip"), (Integer)input.get("port"));
21. log.info("客户端启动（quit退出）");
22. **if** (login((String)input.get("token"))) {
23. log.info("登录成功！");
24. } **else** {
25. log.info("登录失败！");
26. **return**;
27. }
29. sendMsg();
30. }
32. **public** Map<String,Object> input() {
33. HashMap<String, Object> map = **new** HashMap<>();
35. Scanner in = **new** Scanner(System.in);
36. System.out.print("服务器IP：");
37. map.put("ip",in.next());
38. System.out.print("服务器端口：");
39. map.put("port",in.nextInt());
40. System.out.print("口令：");
41. map.put("token",in.next());
43. **return** map;
44. }
46. **public** Boolean login(String token) **throws** IOException {
47. OutputStream outputStream = client.getOutputStream();
48. PrintWriter printWriter = **new** PrintWriter(outputStream);
49. printWriter.println(token);
50. printWriter.flush();
51. log.info("账号口令已发送");
53. **boolean** state = **false**;
54. InputStream inputStream = client.getInputStream();
55. BufferedReader reader = **new** BufferedReader(**new** InputStreamReader(inputStream));
56. **if** (Integer.valueOf(reader.readLine()) == 1 ) {
57. state = **true**;
58. }
60. **return** state;
61. }
63. **public** **void** sendMsg() **throws** IOException {
64. OutputStream outputStream = client.getOutputStream();
65. PrintWriter printWriter = **new** PrintWriter(outputStream);
66. Scanner in = **new** Scanner(System.in);
67. **while** (**true**) {
68. System.out.print("客户端：");
69. String msg = in.nextLine();
70. printWriter.println(msg);
71. printWriter.flush();
72. **if** (msg.equalsIgnoreCase("quit")) {
73. log.info("客户端退出。。。");
74. **break**;
75. }
76. }
77. printWriter.close();
78. outputStream.close();
79. }
80. }
81. **实验结果分析**

**4.1 启动服务端，并输入敏感词（哈哈）**

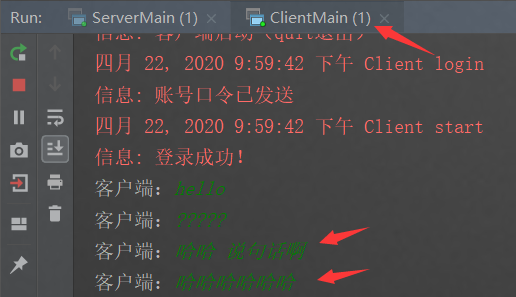
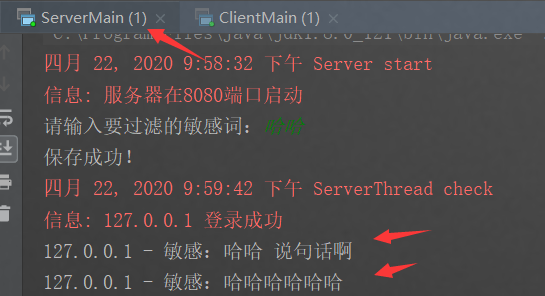


**4.2 启动客户端，尝试登录；同时观察客户端与服务端**

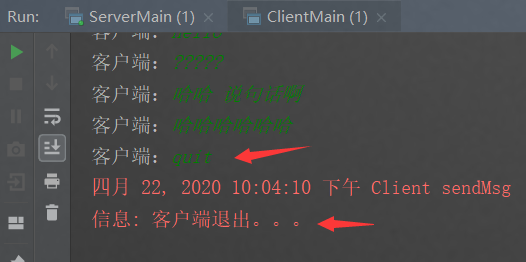
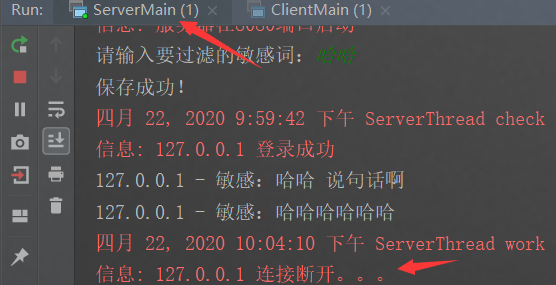
 



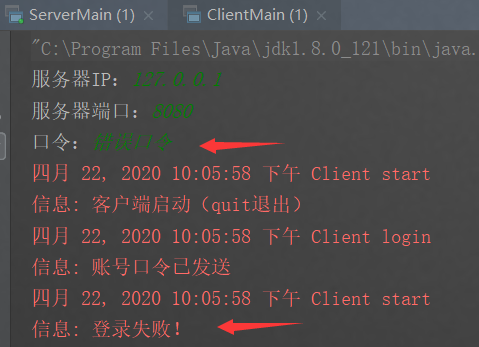
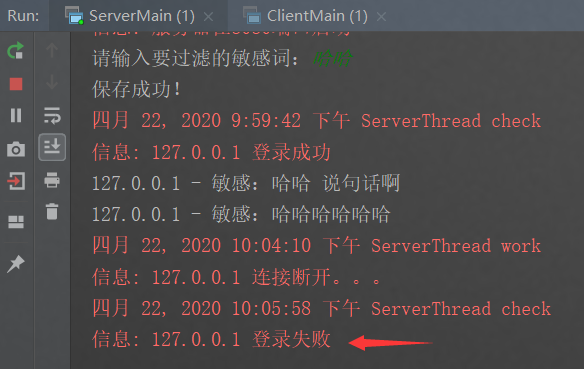
**4.3 客户端向服务端发送消息，服务端进行敏感词校验**

**4.4 客户端退出**

**4.5 演示客户端口令错误的情况**

**五．实验总结及心得体会**