|  |  |  |
| --- | --- | --- |
| 学号： |  |  |
| 姓名： |  |

**《大数据安全与隐私》课程**

**实验报告**

|  |  |
| --- | --- |
| **学号：** |  |
| **姓名：** |  |
| **指导教师：** | 彭延国 |
| **日期：** | 2022-6-10 |

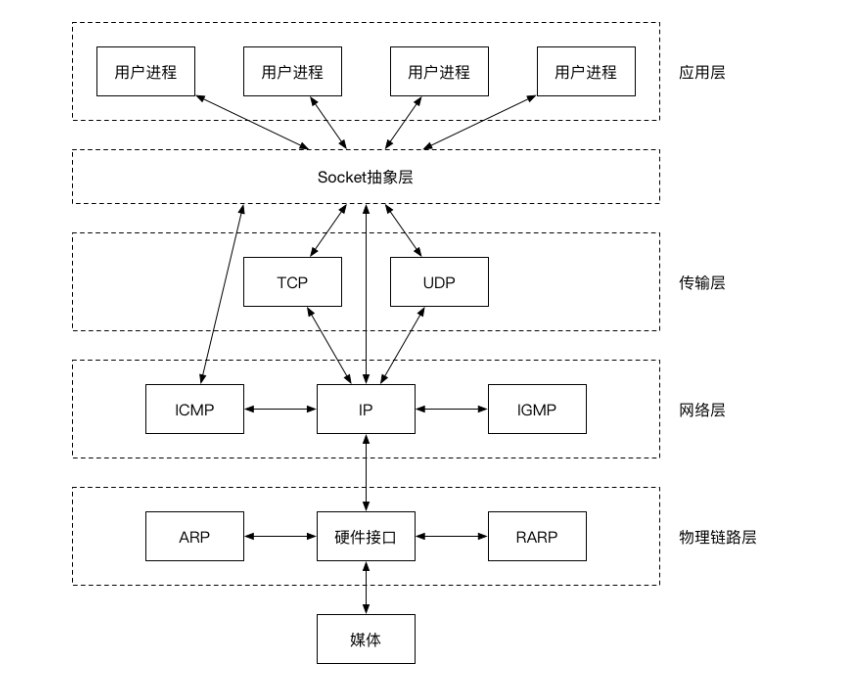
**实验报告撰写要求**

1. 不得抄袭、复制，一经发现取消课程实验成绩；
2. 正文小四字号，固定行距20磅，中文宋体，英文Times New Roman；
3. 每次实验起新页；
4. 实验报告电子档命名规则：“《大数据安全与隐私》实验报告-学号-姓名.doc/.docx/.pdf”；
5. A4纸单面打印，一份；
6. 实验报告作为实验验收的补充，仅提交实验报告未进行线上/线下验收的没有实验部分成绩。
7. 提交时间：2022年6月1日后。具体时间待全部验收后，通过学在西电和微信群通知。

**实验一：现代密码学基础技能（阶梯三）**

1. **实验原理：**

1、Socket通信原理



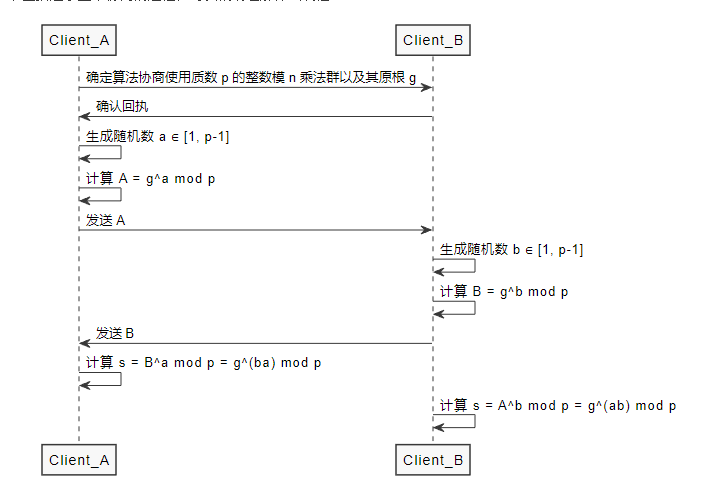
Socket是在应用层和传输层之间的一个抽象层，它把TCP/IP层复杂的操作抽象为几个简单的接口，供应用层调用实现进程在网络中的通信。Socket起源于UNIX，在Unix一切皆文件的思想下，进程间通信就被冠名为文件描述符（file desciptor），Socket是一种“打开—读/写—关闭”模式的实现，服务器和客户端各自维护一个“文件”，在建立连接打开后，可以向文件写入内容供对方读取或者读取对方内容，通讯结束时关闭文件。

在Socket网络通信过程需要分别构建服务端和客户端，服务器与客户端之间通信时，两端都建立了一个Socket对象，然后通过Socket对象对数据进行传输。通常服务器处于一个无限循环，等待客户端的连接。

2、DH密钥协商过程

迪菲－赫尔曼（Diffie–Hellman）密钥协商是在美国密码学家惠特菲尔德·迪菲和马丁·赫尔曼的合作下发明的，发表于 1976 年。它是第一个实用的在非保护信道中创建共享密钥方法。DH 算法可以在一个不安全的信道上建立安全连接，从而解决的不安全信道上信息安全交换的问题。

假设 Client\_A 与 Client\_B 在不安全的信道上交换信息，他们通过 DH 算法协商出一个密钥，具体流程如下:



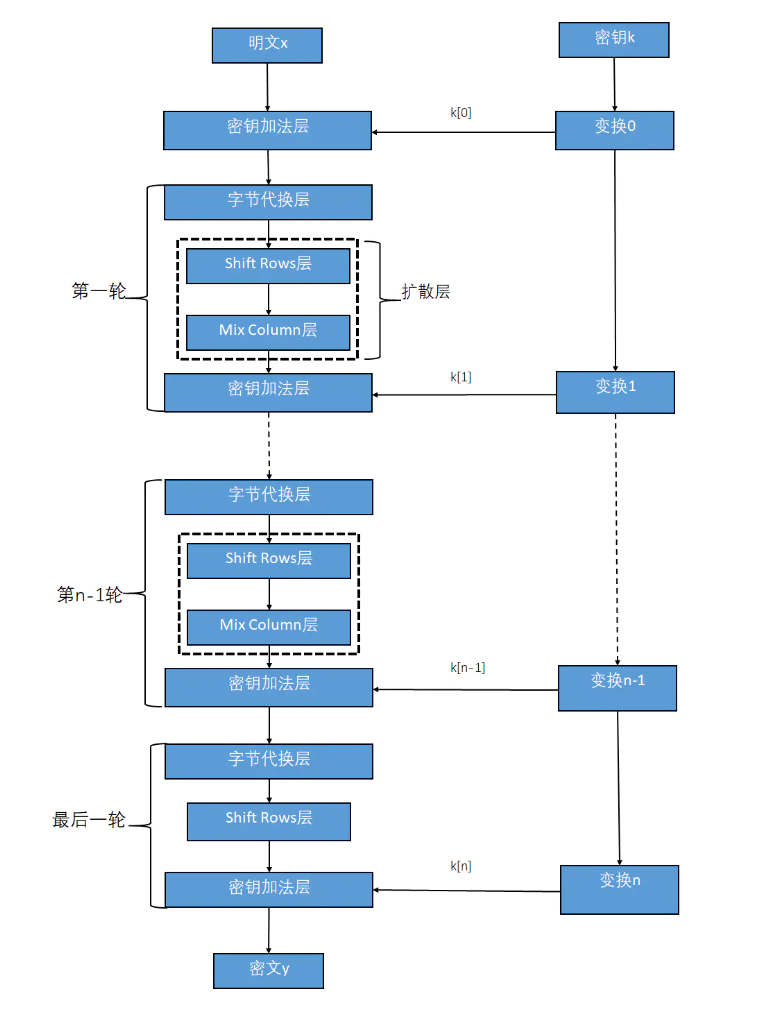
1. Client\_A 与 Client\_B 确定算法协商使用质数 p 的整数模 n 乘法群以及其原根 g
2. Client\_A 生成随机数 ，计算 ，将 A 发送给 Client\_B
3. Client\_B 生成随机数 ，计算 B =，将 B 发送给 Client\_A
4. Client\_A 计算
5. Client\_B 计算

通过上述过程，Client\_A 与 Client\_B 得到了一个安全的共享密钥 s。

3、AES加密过程

高级加密标准(AES,Advanced Encryption Standard)，又称Rijndael加密法，是美国联邦政府采用的一种区块加密标准。这个标准用来替代原先的DES（Data Encryption Standard），已经被多方分析且广为全世界所使用。

严格地说，AES和Rijndael加密法并不完全一样（虽然在实际应用中两者可以互换），因为Rijndael加密法可以支持更大范围的区块和密钥长度：AES的区块长度固定为128比特，密钥长度则可以是128，192或256比特；而Rijndael使用的密钥和区块长度均可以是128，192或256比特。加密过程中使用的密钥是由Rijndael密钥生成方案产生。



AES加密过程在一个4×4的字节矩阵上运作，其初值就是一个明文区块（矩阵中一个元素大小就是明文区块中的一个Byte）。其矩阵的“列数（Row number）”可视情况增加）加密时，各轮AES加密循环（除最后一轮外）均包含4个步骤：

（1）AddRoundKey—矩阵中的每一个字节都与该次回合密钥（round key）做XOR运算；每个子密钥由密钥生成方案产生。

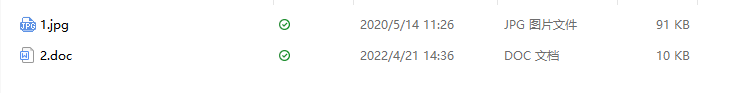
（2）SubBytes—透过一个非线性的替换函数，用查找表的方式把每个字节替换成对应的字节。

（3）ShiftRows—将矩阵中的每个横列进行循环式移位。

（4）、MixColumns—为了充分混合矩阵中各个直行的操作。这个步骤使用线性转换来混合每内联的四个字节。最后一个加密循环中省略MixColumns步骤，而以另一个AddRoundKey取代。

4、实验步骤

（1）确定服务端和客户端运行环境，服务端为腾讯云服务器，系统为Linux CentOS7，客户端为本地主机，现有文件1.jpg及2.doc欲发送至客户端。



客户端使用files文件夹接收待传输文件，初始为空。

（2）服务器开启socket监听，开放10870端口等待客户端连接；之后客户端确立连接端口并发起连接。

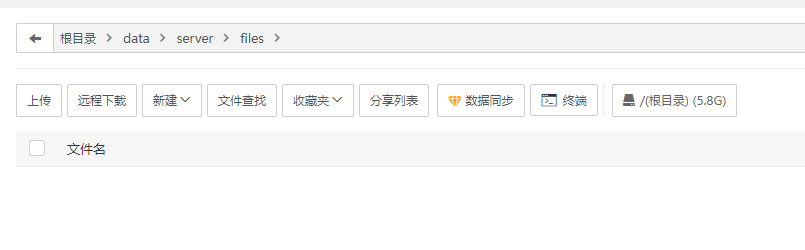
（3）建立连接后两者进行DH回话密钥协商，根据DH密钥生成AES密钥，客户端使用AES密钥加密文件名信息并发送给服务端，服务端使用生成的 AES密钥进行解密。

（4）客户端以二进制流形式读取文件信息，加密后发送给服务端。服务端收到解密后，依据（3）中文件名信息写入文件并保存在files文件夹下。

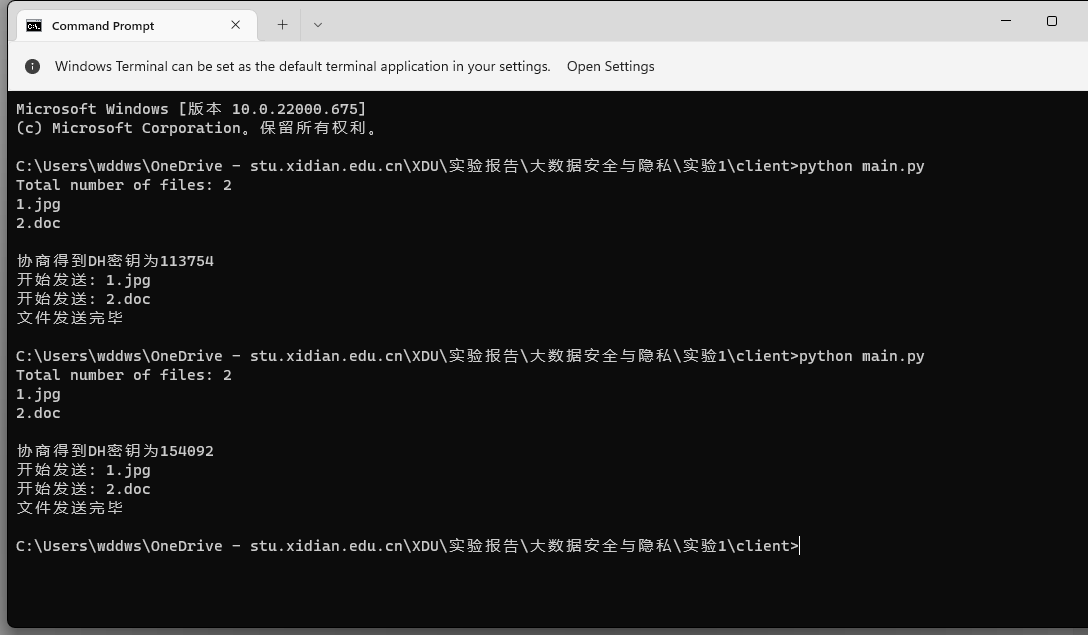
（5）确认文件发送完成后客户端主动向服务端发送close信息中断连接，服务端收到信息自动中断监听并关闭连接。

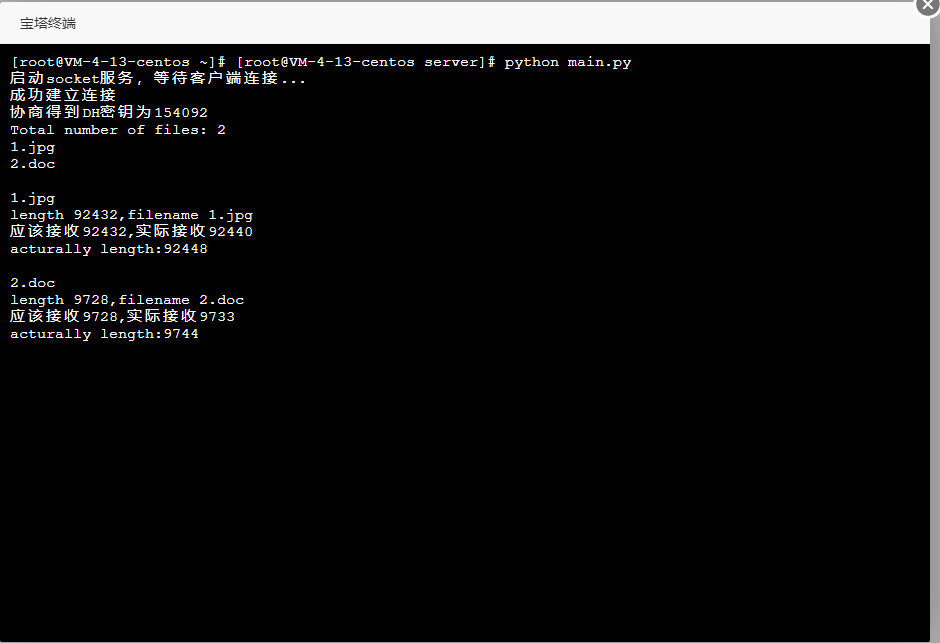
1. **实验结果：**

1、实验开始前进行检查，登录腾讯云服务器，发现服务端未保存文件。

****

2、服务端执行python命令开始监听，随后开启客户端，执行结果如下：





3、运行完毕后检查服务端文件接收情况，结果如下：







传输后图片和doc内容能够正常写入并显示，文件传输成功。

1. **实验总结与心得：**

编写文件加密传输程序时，不同于之前使用txt文件作为加密内容，我在客户端直接对jpg文件和doc文件对应的二进制流进行加密，服务端收到加密二进制流后解密形成明文二进制流，创建并写入对应文件。由于没有多文件格式在真实网络环境下的传输代码编写经历，实验过程较为曲折。

在实际编写程序时，遇到了许多困难，比如我准备通过将文件名信息加密后一并传输的方式，以便服务端确定写入文件的文件类型，在加密过程中难免遇到字符串与二进制流的编码解码问题。

在使用socket进行文件传输时，加密传输一个文件后服务端便停止接收文件，经过调试后采用传输文件时重建连接的方式解决。

实验使用腾讯云服务器作为服务端，但在真实网络环境传输中，总会出现丢包问题，文件传输后的字节总数和传输前不一致，导致AES解密时位数不能满足要求。后来我对服务端收到的加密文件进行二进制补0，补0至满足AES解密的位数要求，测试发现能正确写入文件，并未改变解密后的文件内容。

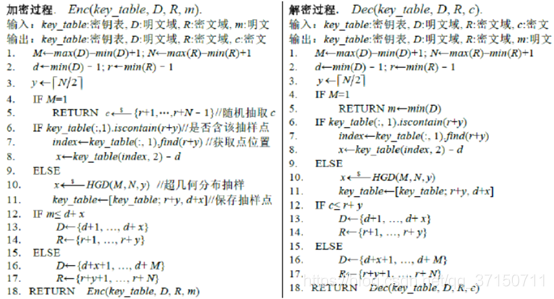
在本次实验中，我加深了对文件加密传输的理解，对分组加密、回话密钥加密算法和消息摘要算法的使用更加熟练，并对真实网络环境中实现文件加密传输的难度有了进一步认识，收获颇丰。

**实验二：数据安全检索（阶梯三）**

1. **实验原理：**
2. 保序加密原理

保序加密(Order-Preserving Encryption)算法：最初是由2009年，Boldyreva等四个人提出，可简称BCLO-09算法，发表论文题目为《Order-Preserving Symmetric Encryption》。

OPE算法保证了经算法加密后明文的大小顺序不变，且密文可以通过密钥表还原为明文，窃听者获取到密文后除了明文的大小顺序无法获取任何有关明文的信息，算法流程如下：



1. Kd树生成及最近邻查询过程

kd树是一个二叉树结构，它的每一个节点记载了(特征坐标，切分轴，指向左枝的指针，指向右枝的指针)。

特征坐标为N维空间中的一个坐标，切分轴r表示当前子树按第r维坐标值进行切分。

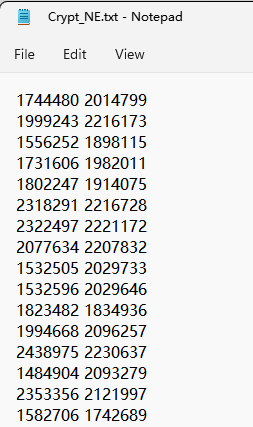
节点的左枝和右枝分别都是 kd 树，并且满足：如果 y 是左枝的一个特征坐标，那么，并且如果 z 是右枝的一个特征坐标，那么。

N维Kd树使用递归构造，构造第r级子树时按照r % N维坐标值对当前子树坐标点进行切分，选取中位值对应的坐标x作为根节点，r % N维坐标值小于x的节点构造左子树，反之则构造右子树。

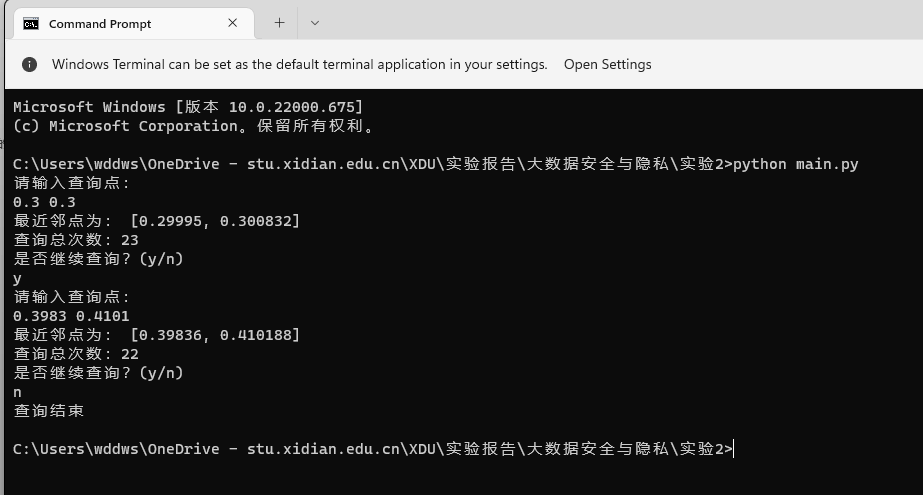
利用Kd树的这一性质可以很方便的进行最近邻查询，先根据查询坐标点的坐标值通过二叉搜索找到与坐标点相近的最末级子树的根节点。以查询坐标为圆心，距该节点距离为半径画圆，若圆与各维度超平面均不交割，则确定根节点为最近邻点，否则递归搜索上一级子树。

1. 实验步骤
2. 读取文件NE.txt并将二维坐标点以列表形式存放。
3. 设定OPE加密映射密文域，对二维数据集进行OPE加密，保存为Crypt\_NE.txt。
4. 将二维数据集存储为kd树，存储过程以递归函数实现，返回一个树形元组，以便实现最近邻查询算法。
5. 编写最近邻查询算法对给定坐标点查询在对应kd树中的最近邻点。
6. 执行程序，并使用坐标点测试查询结果。
7. **实验结果：**

1、明文加密后的部分数据，对比明文可以发现加密数据具有保序性。



2、对数据集进行最近邻查询，发现查询结果符合预期，能正确查找最近邻点。递归执行次数较少，对比简单二叉搜索效率提升较大。



1. **实验总结与心得：**

在实现最近邻查询算法的过程中，需要通过判断以查询坐标点为圆心，当前子树根节点与其距离为半径所构造的圆是否与超平面相交。在这一过程中，需要使用递归回溯判断最近邻点，比较难以理解。在实验中我遇到了递归过程中重复查询某子树、递归返回值异常等问题，最后通过加入异常判断（置inf）、完善递归调用顺序等方法解决了问题，让我对递归算法的实现有了更高认识。

在实验过程中，为验证最近邻查询算法的正确性，我构建了简单二叉搜索算法对kd树进行同步查询。在确定算法正确之后，我进行了简单二叉搜索算法与基于kd树的最近邻搜索算法的时间复杂度对比，相交之下，最近邻搜索算法的时间复杂度大大降低，在数据规模为数十万的情况下有着百倍于简单二叉搜索算法的性能。由此可见，构建适当的索引结构，对于数据的检索是非常必要且有意义的。

通过本次实验，我充分理解了保序加密算法的原理及实现方法，学习了使用kd树存储数据并实现最近邻检索，体会到安全数据库存储及数据检索过程的复杂性和重要性。

**实验三：大数据隐私保护（阶梯三）**

1. **实验原理：**
2. K-匿名原理
3. 隐私信息概念及定义

**标识符**：可以直接确定一个个体，如身份证号、姓名等。

**准标识符集**：可以和外部表链接来识别个体的最小属性集，如邮编、生日、性别等。

**敏感数据**：用户不希望被人知道的数据，如薪水、疾病历史、购买偏好等。

1. K-匿名技术

K-匿名技术可以保证存储在发布数据集中的每条个体记录对于敏感属性不能与其他的K-1个个体相区分，通俗的说就是同一个准标识符集至少要有K条记录，因此观察者无法通过准标识符集链接记录。

K-匿名实施通常是通过概括和隐匿。概括指对数据进一步抽象描述，比如把年龄概括成年龄段。隐匿指隐藏数据的部分信息，比如邮政编码隐藏后几位。这么处理降低了数据的精度，使得每条记录与其他K-1条记录有完全相同的准标识符集，降低了链接攻击所导致的隐私泄露风险。

1. K-匿名作用

K-匿名技术保证了：

* 无法得知特定的人是否在公开的数据中
* 无法确定特定的人是否有某项敏感属性
* 无法确认某条数据对应哪个人

1. K-匿名的缺陷

虽然可以阻止身份信息的公开，但无法防止属性信息的公开，导致无法抵抗同质攻击、背景知识攻击、补充数据攻击等。

同质攻击：K-匿名的敏感数据一样，则只要知道一个用户的准标识符集，就能知道该用户具有这个敏感数据。

背景攻击：通过用户的准标识符集确定用户所在的等价信息表，同时观察者通过背景知识了解到用户没有某一敏感数据，就可以用排除法得到用户的敏感数据。

补充数据攻击：当公开的数据有多种类型，若这些数据的K-匿名方法不同，那么攻击者可以通过关联多种数据推测用户信息。

1. 差分隐私处理原理
2. 差分隐私的目的

假设攻击者具有一定背景知识，通过差分隐私处理，使得对于差别只有一条记录的两个数据集，查询它们获得相同值的概率非常接近。从而使得攻击者无法通过手中的背景知识判断剩余受差分隐私保护者的隐私信息。

1. 差分隐私通常的处理方式

通常在结果上加满足某种分布的噪音，使查询结果随机化。

目前常用的有两种方法，一个是Laplace机制，在查询结果里加入Laplace分布的噪音，适用于数值型输出。另外一个是指数机制，在查询结果里用指数分布来调整概率，适用于非数值型输出。

3、差分隐私存在的缺陷

由于对于背景知识的假设过强，需要在查询结果中加入大量噪声，导致数据的可用性急剧下降。特别对于复杂查询，有时候随机化结果几乎掩盖了真实结果。

4、实验步骤

（1）读入数据并将其转化为DataFrame格式。将数据集内的非数值型数据转化成category类型，提高索引速度。

（2）对数据进行 K 匿名处理，并计算 K 匿名处理后每组的平均年龄，发布为k\_result.csv。

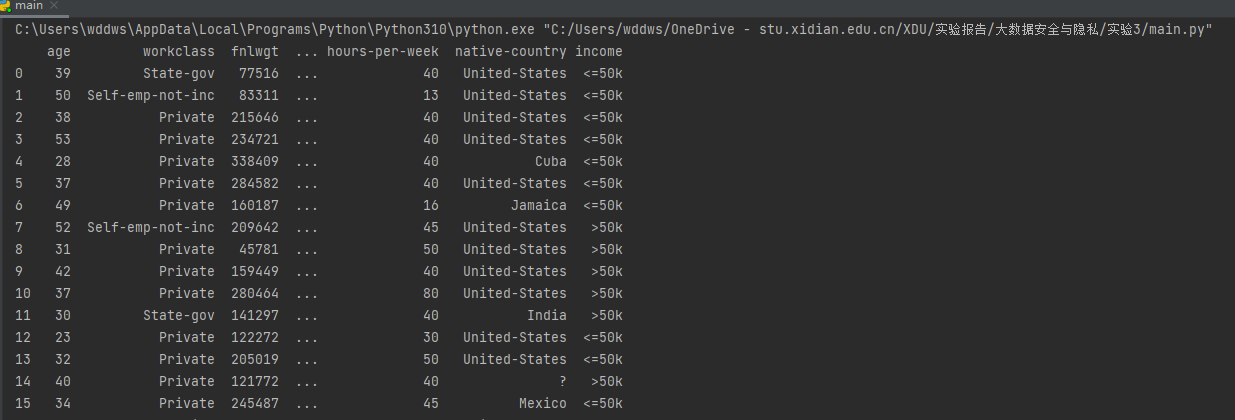
（3）在原数据集基础上删除一条数据，进行 K 匿名处理，并计算每组平均年龄，发布为k\_result\_delete.csv。

（4）执行差分隐私处理，读入k\_result.csv及k\_result\_delete.csv，分别对每组平均年龄加入符合μ=0的拉普拉斯分布噪声（尺度参数b的值由数据规模决定）。

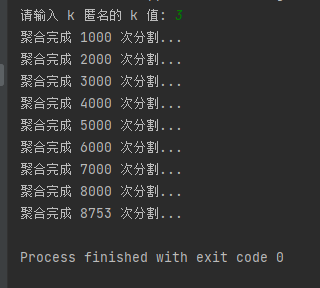
（5）分别计算原数据年龄均值、k匿名处理后的年龄均值、差分隐私处理后的年龄均值，并分析处理过程对年龄均值的影响。

（6）分别对删除后的k匿名发布数据和差分隐私处理数据进行背景攻击，推测删除的年龄均值，分析其与真实年龄的误差。

1. **实验结果：**
   1. 生成Dataframe数据

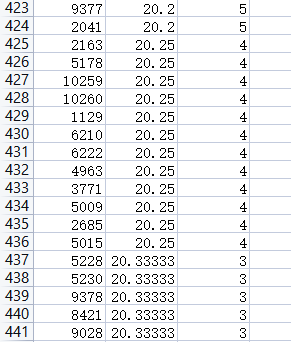


* 1. 执行main.py进行k匿名处理。

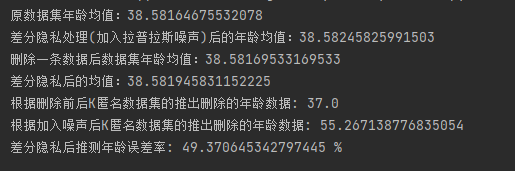


* 1. 发布聚合平均年龄数据集。

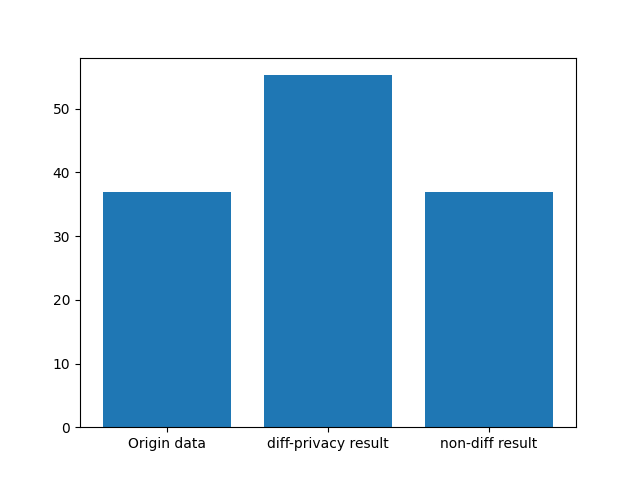




* 1. 进行差分隐私处理，计算数据集内的平均年龄，对比真实年龄数据可知，K匿名后平均年龄数据依然可用。根据数据规模对数据进行差分隐私处理，计算平均年龄可知，差分隐私后平均年龄变化很小，数据依然可用。



5、分别对差分隐私处理前后删除一条数据的数据集和原数据集进行背景攻击，并推测删除数据的年龄，发现差分隐私处理前推测准确，处理后推测误差为49.37%，误差较大。得出结论：差分隐私处理能有效对抗背景攻击。下图为原删除数据对应年龄、差分隐私处理后推测平均年龄以及处理前推测平均年龄之间的对比。



1. **实验总结与心得：**

在这次实验过程中，处理这样记录类型多，数据规模大的数据集对我来说是一个很大的挑战。在实现k匿名的过程中，如何对不同跨度的记录类型进行泛化，如何对数据集进行合理分割，如何判断处理数据满足l多样性和k匿名要求。这些问题的解决加深了我对数据处理过程的认识，提高了我对numpy和pandas数据处理库运用的理解。

在差分隐私过程中，使用拉普拉斯噪声处理数据让我认识到，个体数据加入的微小噪声在较大数据集上往往能取得意想不到的奇效。使用拉普拉斯分布的噪声对原数据集处理后，背景攻击对改动数据的猜测效果误差急遽增大，这种保护隐私的处理策略给了我很多启发。

通过本次实验，我不仅学会了如何使用K匿名和差分隐私处理方法保护用户隐私，更是加深了我对大规模数据安全保护的认识，增强了我对大规模数据集的分析能力，使我受益良多。