创新创业实践

Project 6

实验报告

姓名 迟曼

学号 202200460070

学院 网络空间安全

专业 网络空间安全

目录

一、实验目的

Project 6: Google Password Checkup 验证

来自刘巍然老师的报告 google password checkup，参考论文 https://eprint.iacr.org/2019/723.pdf 的 section 3.1 ，也即 Figure2 中展示的协议，尝试实现该协议，（编程语言不限）。

二、实验原理

DDH-based Private Intersection-Sum（基于 DDH 的私密交集求和）是一种密码学协议，用于解决两方或多方在保护各自数据隐私的前提下，计算交集元素的关联权重之和的问题。其核心思想基于 Decisional Diffie-Hellman（DDH）假设和隐私集合求交（PSI）技术，通过结合代数运算和密码学工具实现安全计算。根据论文 Section 3.1(Figure 2)中的协议描述，以下将从数学角度推导和分析该协议，包括形式化表示、安全性和正确性证明。

（一）协议形式化表示

1.协议参数定义

G：素数阶 q 的循环群，生成元为 g

$H: U \to G$：哈希函数（建模为随机预言机）

(pk, sk)：加法同态加密方案的密钥对

$V = \{v_i\}_{i=1}^{m_1}$：P1 的标识符集合

$W = \{(w_j, t_j)\}_{j=1}^{m_2}$：P2 的标识符-值对集合

$k_1, k_2 \in Z_q^*$：双方随机选择的私钥

2.协议流程形式化描述

Setup:

P2 生成 $(pk, sk) \leftarrow AGen(\lambda)$　　P2 发送 pk 给 P1

Round 1 (P1 → P2):

$\forall v_i \in V$:

计算 $A_i = H(v_i)^{k_1}$发送 $\{A_i\}_{i=1}^{m_1}$（随机排列）

Round 2 (P2 → P1):

$\forall A_i \in \{A_i\}$:

计算 $B_i = A_i^{k_2} = H(v_i)^{k_1 k_2}$，发送 $\{B_i\}_{i=1}^{m_1}$（随机排列）

$\forall (w_j, t_j) \in W$:

计算 $C_j = H(w_j)^{k_2}$，计算 $D_j = Enc(pk, t_j)$

发送 $\{(C_j, D_j)\}_{j=1}^{m_2}$（随机排列）

Round 3 (P1):

$\forall (C_j, D_j)$：

计算 $E_j = C_j^{k_1} = H(w_j)^{k_1 k_2}$，找出 $J = \{j \mid E_j \in \{B_i\}\}$

计算 $S = \sum_{j\in J} D_j$（同态求和），发送 $Refresh(S)$ 给 P2

Output (P2):

解密 $S' = Dec(sk, Refresh(S))$

输出 S' 和 $|J|$

（二）数学推导与分析

1. 正确性证明

定理：协议正确计算交集和 $S = \sum_{j\in J}^{t_j}$ ，其中 $J = \{j \mid w_j \in V\}$

证明：

交集标识符满足：

$$w_j \in V \cap W \Leftrightarrow H(w_j)^{k_1 k_2} \in \{H(v_i)^{k_1 k_2}\}$$

同态加密保持加法同态性：

$$Dec(sk, \sum_{j\in J} Enc(pk, t_j)) = \sum_{j\in J} t_j$$

Refresh 操作不影响解密结果：

$$Dec(sk, Refresh(S)) = Dec(sk, S)$$

2.安全性分析（半诚实模型）

定理 1：P1 的视图可模拟，仅泄露 m_2 和 $|J|$

模拟器构造：

输入：$\{v_i\}, m_2, |J|$

模拟：

生成随机 $k_1 \in Z_q^*$ 生成 $\{A_i = g^{r_i}\}_{i=1}^{m_1}$（$ri$ 随机）

生成 $\{B_i = g^{r_i}\}_{i=1}^{m_1}$（$si$ 随机） 生成 $\{C_i = g^{r_i}, Enc(0)\}_{i=1}^{m_2}$

输出模拟视图

不可区分性：基于 DDH 假设，真实分布与模拟分布在计算上不可区分

定理 2：P2 的视图可模拟，仅泄露 m_1 和 S

模拟器构造：

输入：$\{(w_j, t_j)\}, m_1, S$

模拟：

生成随机 $k_2 \in Z_q^*$ 生成 $\{A_i = g^{r_i}\}_{i=1}^{m_1}$（$r_i$ 随机）

计算 $Enc(pk, S)$输出模拟视图

不可区分性：基于 DDH 和 HE 的语义安全

3.安全假设

1.DDH 假设：

$$\left(g, g^a, g^b, g^{ab}\right) \approx_c \left(g, g^a, g^b, g^c\right)$$

2.随机预言机模型：

H 的行为等同于随机函数

3.同态加密安全性：

HE 方案满足 IND-CPA 安全

（三）协议扩展的数学表示

1.阈值变体

P1 在 Round 3 添加判断：

$$if\ \ |J| < \tau\ then\ abort$$

2.反向变体（Appendix G）

P1 在 Round 3 发送$\{Enc(pk, t_j + r_j)\}$替代直接求和

P2 计算：

$$S' = \sum_{j \in J} \left(Enc\left(pk, t_j + r_j\right)\right) - \sum_{j \in J} r_j$$

3.分段输入

将 V 划分为 k 个不相交子集 $V_1,\ldots,V_k$，对每个子集独立执行协议，最终获得 $\{(S_i,|J_i|)\}_{i=1}^{k}$

（四）效率分析

1.计算复杂度

操作	数量
群指数运算	$O(m_1+m_2)$
同态加密	$O(m_2)$
同态加法	$O(1)$

2.通信复杂度

阶　段	通信量
Round 1	$O(m_1)$ 群元素
Round 2	$O(m_1+m_2)$ 群元素 $+O(m_2)$ 密文
Round 3	$O(1)$ 密文

（五）安全边界与局限性

半诚实模型限制无法防止恶意行为（如篡改输入），可通过零知识证明扩展至恶意模型，可通过差分隐私添加噪声，$S\sim=S+Lap(\epsilon\Delta)$；在关联值隐私层面，当 $|J|=1$ 时，S 直接泄露 t_j。

四、代码分析

（一）初始化与密钥生成

双方选择相同的椭圆曲线（NIST P-256），P2 生成 Paillier 同态加密密钥对（公钥用于加密，私钥用于最终解密），P1 和 P2 各自生成一个随机的椭圆曲线私钥（k1 和 k2）。

```python
class Party:
    def __init__(self, items):
        """
        初始化参与方
        P1: items = [id1, id2, ...] (标识符列表)
        P2: items = [(id1, value1), (id2, value2), ...] (标识符-值对)
        """
        self.items = items


def ddh_intersection_sum(p1, p2):
    """
    实现Figure 2中的Private Intersection-Sum协议
    返回交集关联值的和
    """
    # ===== 1. 参数设置 =====
    n = ORDER  # 曲线阶

    # ===== 2. Setup阶段 =====
    # P2生成Paillier密钥对
    public_key, private_key = phe.generate_paillier_keypair(n_length=1024)

    # 双方生成椭圆曲线私钥
    k1 = random.randint(1, n - 1)  # P1私钥
    k2 = random.randint(1, n - 1)  # P2私钥
```

（二）第一轮通信（P1→P2）

P1 对其每个标识符进行哈希处理，将其映射到椭圆曲线上的一个点。P1 用自己的私钥k_1对这些点进行标量乘法（即点乘）。P1 将这些处理后的点转换为字节格式并发送给 P2。

```python
    # ===== 3. Round 1 (P1 → P2) =====
    p1_to_p2 = []  # P1发送给P2的数据

    for identifier in p1.items:
        # 将标识符哈希到曲线上的点
        H_point = _hash_to_point(identifier)

        # 计算点乘: H(identifier)^k1
        k1H_point = _scalar_multiply(H_point, k1)

        # 将点转换为字节形式进行传输
        p1_to_p2.append(point_to_bytes(k1H_point))
```

（三）第二轮通信（P2→P1）

P2 收到 P1 的点后，用自己的私钥k_2再次进行标量乘法（相当于计算$H(id)^\wedge(k_1 * k_2)$），形成用于交集测试的集合 Z。P2 对自己的每个标识符-值对做以下操作：

1.将标识符哈希映射到椭圆曲线点。

2.用私钥k_2进行标量乘法。

3.用 Paillier 公钥加密关联的值。

P2 将处理后的点$(H(id)^{\wedge}k_2)$和加密值打包，并随机打乱顺序后发送给 P1。

```
# ===== 4. Round 2 (P2 → P1) =====
# 4.1 计算Z = {H(v_i)^k1k₂}
z_set = []  # 交集测试集

for point_bytes in p1_to_p2:
    # 将字节反序列化为点
    point = bytes_to_point(point_bytes)

    # 计算点乘: (H(v_i)^k1)^k₂
    k1k2_point = _scalar_multiply(point, k2)
    z_set.append(point_to_bytes(k1k2_point))

# 4.2 计算P2的集合: (H(w_j)^k₂, Enc(t_j))
p2_to_p1_set = []  # P2发送给P1的数据

for identifier, value in p2.items:
    # 计算H(w_j)^k₂
    H_point = _hash_to_point(identifier)
    k2H_point = _scalar_multiply(H_point, k2)

    # 加密关联值
    enc_value = public_key.encrypt(value)

    # 存储点和加密值
    p2_to_p1_set.append((point_to_bytes(k2H_point), enc_value))

# 打乱顺序发送
random.shuffle(z_set)
random.shuffle(p2_to_p1_set)
```

（四）第三轮处理（P1 本地计算）与结果输出（P2 解密）

P1 收到 P2 发送的数据后：

对每个收到的点$(H(id)^{\wedge}k_2)$用自己的私钥k_1进行标量乘法（也得到$H(id)^{\wedge}(k_1 * k_2)$）。检查这个结果是否存在于之前 P2 发来的集合 Z 中（即判断是否为交集元素）。如果是交集元素，则使用 Paillier 同态加法将对应的加密值累加到一个总和中。P1 将累加得到的加密总和发送给 P2，P2 用自己的 Paillier 私钥解密这个总和，得到最终的交集关联值之和。

```python
    # ===== 5. Round 3 (P1) =====
    # 5.1 计算交集
    intersection_sum = public_key.encrypt(0)  # 初始化为加密的0

    # 优化：创建z_set的哈希索引
    z_index = set(z_set)

    for k2H_point_bytes, enc_value in p2_to_p1_set:
        # 计算H(w_j)^k1k₂
        k2H_point = bytes_to_point(k2H_point_bytes)
        k1k2_point = point_to_bytes(_scalar_multiply(k2H_point, k1))

        # 检查是否在Z集合中
        if k1k2_point in z_index:
            # 同态加法
            intersection_sum += enc_value

    # ===== 6. Output (P2) =====
    return private_key.decrypt(intersection_sum)
```

五、总结与思考

本次实验深入理解了基于 DDH 假设的私密交集求和协议在实际系统中的工程实现。通过编程实现论文中的密码学协议，我们亲身体验了理论设计与工程落地之间的关键差异。椭圆曲线参数的选择直接影响协议安全性，最终采用 NIST P-256 曲线满足安全强度需求；而 Paillier 同态加密方案的密钥长度优化则成为性能瓶颈突破点，测试发现 2048 位密钥能在安全性和计算开销间取得较好平衡。

在协议执行过程中，第二轮通信时对数据集的洗牌处理来严格保证随机性，避免攻击者通过时序分析推测元素关联性。此外，椭圆曲线点与字节序列的高效转换也需特殊处理，采用压缩坐标表示后通信量降低 50%，有效缓解了原协议中$O(m_1+m_2)$通信复杂度带来的压力。

在半诚实模型下，通过模拟器构造证明了视图的不可区分性，但当交集元素唯一时（$|J|=1$），关联值的隐私泄露风险依然存在。这促使我们在实现中添加了差分隐私保护层，引入噪声后的输出显著提升了数据隐私保障，但是造成了聚合结果的微小偏差。

实践中我认识到哈希函数到椭圆曲线点的映射需防范无效点异常，通过哈希后取模 q 再乘基点的方案确保了点有效性；同态密文刷新操作则解决了 Paillier 加密的密文膨胀问题，使最终传输的密文大小恒定在 4096 位。这些细节处理虽未出现在理论描述中，却是保证协议鲁棒性的关键。