

密码学导论课程实践

姓名：木拉提·阿力木别克 PB2030831

目录

1	作品题	1
1.1	工作流程	1
1.2	函数功能	1
1.3	单表代换加密和解密与辅助破译单表代换密文	3
1.4	PPT 上示例运行演示：	4
1.5	实验反思与总结	5
2	文献题	6
2.1	研究目标	6
2.2	主要贡献	6
2.3	思考	7

□

1 作品题

我选择了单人作品题目 1：“单表代换辅助工具”
本次完成的单表代换辅助工具代码实现了单表代换密码的加密、解密及辅助破译功能，基于频率分析和常见词匹配的唯一密文攻击，以下对其功能的讲解：

1.1 工作流程

初始化密钥 (全未知)→ 分析密文字母频率 → 生成与标准英语频率的映射建议 → 显示部分解密结果 → 尝试匹配已知单词模式 → 允许用户选择应用建议或手动输入替换
(重复上述步骤直到完全破译)

1.2 函数功能

1. 主要数据结构

- CipherKey : 存储替换表和已知替换数量

- `LetterStats` : 存储字母出现次数和频率
- `Suggestion` : 存储替换建议

2. 核心功能函数

- `initializeKey(CipherKey *key)`
 - 作用: 初始化密钥结构体, 将所有替换和反向替换设置为未知 ('i'), 已知位置数设为 0
- `encrypt(const char *plaintext, char *ciphertext, CipherKey *key)`
 - 作用: 使用当前密钥加密明文, 遍历明文字母, 根据替换表转换为密文, 非字母字符保持不变
- `decrypt(const char *ciphertext, char *plaintext, CipherKey *key)`
 - 作用: 使用当前密钥解密密文, 遍历密文字母, 根据反向替换表转换为明文, 非字母字符保持不变
- `generateRandomKey(CipherKey *key)`
 - 作用: 生成随机替换密钥, 使用 Fisher-Yates 洗牌算法随机排列字母表创建替换表

3. 密码分析函数

- `analyzeFrequency(const char *text, LetterStats stats[])`
 - 作用: 分析文本的字母频率
- `printFrequencyAnalysis(LetterStats stats[])`
 - 作用: 打印频率分析结果
- `suggestMappings(LetterStats cipher_stats[], Suggestion suggestions[])`
 - 作用: 将密文字母按频率排序, 与标准英语频率排序匹配, 生成替换建议映射
- `printSuggestions(Suggestion suggestions[])`
 - 作用: 打印替换建议, 显示所有可能的密文到明文字母映射及其匹配差异度

4. 交互破译函数

- `applySuggestion(CipherKey *key, char cipher_char, char plain_char)`
 - 作用: 单个字母替换, 更新替换表和反向替换表, 检查冲突
- `partialDecrypt(const char *ciphertext, char *partial, CipherKey *key)`
 - 作用: 部分解密密文 (已知字母显示为小写, 未知保持大写)
- `findPossibleWords(const char *partial, CipherKey *key)`
 - 作用: 查找可能的单词匹配, 在部分解密文本中识别可能与常见短词匹配的模式, 并提出新替换建议
- `interactiveCrack(const char *ciphertext)`
 - 作用: 显示交互破译页面

5. 辅助功能

- 主菜单 (`main()` 中的 switch-case) 提供加密、解密、生成随机密钥、破译密文等功能的用户界面

1.3 单表代换加密和解密与辅助破译单表代换密文

单表代换加密过程

- `initializeKey(&key)` : 初始化替换表, 所有字母映射为未知 ('?')
- `generateRandomKey(&key)` :
 - 随机生成一组替换规则或者使用 Fisher-Yates 洗牌算法
- `encrypt(plaintext, ciphertext, &key)` :
 - 遍历明文字母, 查询替换表 `key.substitution`, 若存在映射则替换, 否则保留原字母。非字母字符 (空格/标点) 保持不变

单表代换解密过程

- `decrypt(ciphertext, decrypted, &key)` :
 - 遍历密文字母, 查询反向替换表 `key.reverse_sub`, 若存在反向映射则还原, 否则保留原字母。非字母字符保持不变

唯密文攻击方法

1. 字母频率分析

- `analyzeFrequency(ciphertext, stats)` :
 - 统计字母频率, 对比英语标准频率。执行 `suggestMappings(stats, suggestions)`, 通过密文字母频率排序和英语字母排序生成建议映射

2. 上下文分析

- `findPossibleWords(partial, &key)` :
 - 识别部分解密模式 (如 T_E) 匹配常见单词 (建议 _→H 得到 THE), 检查高频连接 (TH/QU/ING 等)

3. 交互式验证

- `applySuggestion(&key, cipher_char, plain_char)` :
 - 用户指定映射, 更新双向替换表并检查冲突
- `partialDecrypt(ciphertext, partial, &key)` :
 - 已知映射显示为小写, 未知字母保持大写。
- `interactiveCrack(ciphertext)` :
 - 主循环提供选项:
 1. 应用频率建议、手动输入映射、显示当前密钥状态、完成破译

破译工作流程

1. 初始化全未知密钥 `initializeKey()`
2. 运行 `analyzeFrequency()` 获取统计特征，通过 `suggestMappings()` 生成初始假设，使用 `partialDecrypt()` 显示当前解密状态
3. 循环执行：
 - 用户选择自动建议或手动输入
 - `findPossibleWords()` 识别单词模式
 - 当已知映射足够时终止循环

1.4 PPT 上示例运行演示：

运行程序，按照指示输入“5”，对 PPT 示例进行分析，得到：频率分析、替换建议和下一步的操作

```
替换建议 (密文字母 -> 明文字母):
N -> E (差异: 1.98%)
O -> T (差异: 0.49%)
S -> A (差异: 0.13%)
Z -> O (差异: 0.01%)
L -> I (差异: 0.32%)
C -> N (差异: 0.28%)
Q -> S (差异: 0.14%)
F -> H (差异: 0.20%)
J -> R (差异: 0.31%)
R -> D (差异: 0.23%)
G -> L (差异: 0.18%)
B -> C (差异: 0.36%)
E -> U (差异: 0.31%)
X -> M (差异: 0.04%)
W -> W (差异: 0.09%)
K -> F (差异: 0.13%)
```

图 1: 生成替换建议

字母	计数	频率	英语频率
A	2	0.35%	8.17%
B	18	3.15%	1.49%
C	37	6.47%	2.78%
D	10	1.75%	4.25%
E	14	2.45%	12.70%
F	36	6.29%	2.23%
G	22	3.85%	2.02%
H	9	1.57%	6.09%
I	0	0.00%	6.97%
J	36	6.29%	0.15%
K	12	2.10%	0.77%
L	38	6.64%	4.03%
M	4	0.70%	2.41%
N	84	14.69%	6.75%
O	10	1.75%	0.54%

图 2: 对各字母进行频率分析

```
部分解密结果:
HZSRNQC KLYY WQC FLO MFLWF OL ZQDN NSOZND WSKN LJ XZSRBJNF,WZSXZ GQV ZQH
SF ZSC ZLECN SF CQOSRRN JLV,WZSOZND FLFN HNFNOJQONB.Q CSFYRN BLGNCOSX CEK
CQGN OQPRN,FNDNJ OQMSFY ZSC GNQRC WSOZ LOZND GNGPNJC,GEZ RNCC PJSFYSFY
NQC WZSXZ OZN JNKLJG HJLDSBNC KLJ SOC KQDLEJNB GNGPNJC.ZN HQCCNB ONF ZLEJ

可能的单词匹配:
未找到明显的单词匹配。尝试分析字母频率。

选择操作:
1. 应用替换建议
2. 手动输入替换
3. 显示当前密钥
4. 完成破译
选择: 1
```

图 3: 部分解密结果与下一步操作

选择应用替换建议后我们可以得到替换后的文本，同时程序和词库分析后进一步给出很多可能的单词匹配。同时也会给出 b 分解密结果。

'WINE' 可能匹配 'PAST'
建议: W → P
建议: N → A
建议: F → S
'WINE' 可能匹配 'THAT'
建议: W → T
建议: N → H
建议: F → A
'WINE' 可能匹配 'WHAT'
建议: W → W
建议: N → H
建议: F → A
'Qt' 可能匹配 'AT'
建议: Q → A
'Qt' 可能匹配 'IT'
建议: Q → I
'tL' 可能匹配 'TO'
建议: L → O
'Qt' 可能匹配 'AT'
建议: Q → A
'Qt' 可能匹配 'IT'
建议: Q → I
'tL' 可能匹配 'TO'
建议: L → O
'wzaxZ' 可能匹配 'AGAIN'

图 5: 对各字母进行频率分析

图 6: 部分解密结果

1.5 实验反思与总结

- 5

2 文献题

文献阅读: Quantum Cryptography in 5G Networks: A Comprehensive Overview. (5G 网络中的量子密码学: 全面概述), 发布日期 2023 年 8 月 28 日

2.1 研究目标

本论文聚焦于量子密码学在 5G 网络中的应用, 旨在解决 5G 网络面临的安全挑战, 尤其是量子计算对传统加密技术的威胁。具体目标包括:

(a) **分析 5G 网络安全漏洞:**

研究 5G 网络架构 (如前传、中传、回传) 在量子计算时代的安全风险, 包括传统加密算法 (如 RSA、椭圆曲线密码) 可能被量子算法 (如 Shor 算法、Grover 算法) 破解的隐患。

(b) **验证量子密码学的可行性:**

探讨量子密钥分发 (QKD) 和后量子密码学 (PQC) 如何提升 5G 网络的安全性, 特别是通过 QKD 的信息论安全特性和 PQC 的抗量子算法, 实现密钥分发和通信加密的量子抗性。

(c) **整合量子技术与现有框架:**

研究 QKD 与 PQC 如何融入 5G 现有的安全框架 (如 IPsec、MACsec、TLS), 并设计基于 FPGA 的加密器以满足 5G 的低延迟、高吞吐量需求。

(d) **推动标准化与实际部署:**

分析 QKD 和 PQC 的标准化进展 (如 ETSI、ITU-T 标准), 并通过实际案例 (如马德里量子网络、5G 前传网络测试) 验证技术的落地可行性。

2.2 主要贡献

- 2.1 5G 网络安全架构与挑战分析 - **网络分段安全评估**: 详细分析了 5G 各网络段的安全需求:

- **前传网络 (Fronthaul)**: 基于 eCPRI 接口的低延迟特性, 提出使用 MACsec/IPsec 结合 QKD 密钥加密, 解决光传输中的窃听风险。
- **中传与回传网络 (Midhaul/Backhaul)**: 强调 IPsec 和 TLS 的量子安全增强, 通过 PQC 算法替换传统公钥加密。
- **核心网 (5GC)**: 讨论 NFV 和 SDN 引入的新攻击面, 提出 QKD 与 PQC 在网络切片和边缘计算中的应用方案。

- **量子计算威胁建模**: 量化了量子算法对 5G 认证协议 (如 5G AKA) 的威胁, 指出 128 位密钥在 Grover 算法下的安全强度下降至 64 位, 需通过 256 位密钥或 QKD/PQC 加固。

- 2.2 量子密钥分发 (QKD) 技术体系 - **QKD 原理与协议**: 详细解析 BB84 协议、连续变量 QKD (CV-QKD) 的技术细节, 包括密钥生成、筛选、纠错和隐私放大流程, 并讨论了实际限制 (如光纤损耗导致的密钥速率随距离下降)。

- **QKD 网络架构**: 提出基于可信中继和多跳中继的 QKD 网络模型, 支持跨域密钥分发, 并通过 SDN 控制器实现网络管理与密钥调度 (如马德里量子网络案例)。

- **与现有框架整合**:

- **IPsec/VPN**: 设计 QKD 支持的 IPsec 隧道, 通过快速重密钥 (如每秒 40 次密钥更新) 提升安全性, 减少传统 DH 算法的计算开销。
- **MACsec**: 利用 QKD 密钥实现链路层加密, 结合 FPGA 硬件加速 (如 Virtex UltraScale+ 芯片实现 200 Gbps 吞吐量), 满足 5G 前传的低延迟要求 (<100 s)。
- **2.3 后量子密码学(PQC)方案 - 算法评估与标准化**: 对比分析 NIST 第四轮候选算法 (如 CRYSTALS-Kyber、Sphincs+), 指出格基算法 (Lattice-based) 在计算效率和密钥长度上的优势, 适合 5G 终端设备部署。
 - **5G 应用场景**:
 - **认证与密钥协商**: 在 5G AKA 协议中引入 PQC 签名算法, 替代易受量子攻击的 ECDSA。
 - **TLS/SSH 增强**: 通过混合模式 (传统算法 +PQC) 实现向后兼容, 如亚马逊 s2n 库的 TLS 1.2 改造案例。
 - **性能优化**: 基于 FPGA 的 PQC 算法实现 (如 AES-GCM 加速), 在 10Gbps 速率下延迟低于 350ns, 满足 5G 核心网的高吞吐量需求。
- **2.4 硬件实现与标准化 - FPGA 加密器设计**: 提出三种架构模型 (CPU 加速、独立 FPGA、混合架构), 对比其吞吐量和延迟, 证明 FPGA 在并行处理加密算法 (如 AES、SHA-3) 中的优势, 最高可达 482 Gbps 吞吐量。
 - **标准化进展**: 梳理 ETSI 004/014 标准对 QKD 应用接口的定义, 以及 ITU-T 对 QKD 网络管理的规范, 推动量子安全成为 5G 标准的一部分。

2.3 思考

- **3.1 QKD 与 PQC 的互补性 - 场景适配**: QKD 适合静态、高安全需求的链路 (如核心网骨干), 而 PQC 更适合动态、移动场景 (如终端接入)。两者结合可构建 “量子安全层”, 覆盖 5G 全网络。
 - **混合加密策略**: 短期内采用 “QKD 生成密钥 +PQC 保护控制信道” 的混合模式, 长期逐步过渡到全量子安全架构, 避免单一技术依赖。
- **3.2 标准化与产业落地挑战 - 跨厂商兼容性**: QKD 设备 (如 ID Quantique 的 Centauris 系列) 需统一密钥管理接口 (如 KMS 协议), 解决不同厂商间的互操作性问题。
 - **成本与功耗**: FPGA 和光子器件的成本仍是大规模部署的障碍, 需推动 ASIC 化 (如英特尔 Agilex FPGA 的 IPsec 硬核心) 以降低功耗和成本。
- **3.3 未来研究方向 - 量子中继与长距离传输**: 研究基于纠缠的量子中继器, 突破现有 QKD 的距离限制 (当前约 150 km 光纤), 实现广域网级量子安全。
 - **边缘计算与物联网**: 将 QKD/PQC 嵌入 5G 边缘节点 (如 MEC 服务器), 保护物联网设备的实时通信, 如自动驾驶中的车联网安全。
 - **抗量子攻击的安全协议**: 重新设计 5G 核心协议 (如 NGAP、GTP-U), 从底层集成量子安全机制, 而非事后加固。