**开发手册**

**1.概述**

本项目是基于Python 3和Tkinter库实现的S-AES算法图形化工具。项目设计遵循关注点分离原则，将用户界面(UI)与核心加密逻辑(Core)分离，代码结构清晰，易于维护。

**2.架构设计**

项目采用分层架构，主要分为以下层次：

**表现层**

文件:gui/main\_window.py,gui/tabs/\*.py

职责:构建和管理所有UI组件（主窗口、选项卡、按钮），处理用户输入事件，调用逻辑接口层执行核心任务，并展示结果。

**逻辑层**

文件:core/saes\_interface.py

职责:充当UI与算法核心之间的桥梁。实现作业要求的上层功能（如ASCII处理、双重/三重加密、CBC模式、中间相遇攻击），它负责调用saes\_core中的原子操作来组合成复杂功能。

**算法核心层**

文件:core/saes\_core.py

职责:S-AES算法的纯粹数学实现。包含所有S-AES标准定义的常量（S盒、逆S盒、RCON）、状态矩阵操作、密钥扩展、轮函数以及基础的16位encrypt和decrypt函数。

**工具层**

文件:utils/validators.py

职责:提供通用的辅助函数，例如validate\_binary用于校验用户输入的合法性。

**3.模块详解**

**main.py**

程序主入口。初始化Tkinter根窗口和SAESGUI类实例，启动事件循环。

**gui/main\_window.py**

初始化主窗口，创建ttk.Notebook(选项卡控件)，并加载gui/tabs/目录下的所有功能选项卡。

**gui/tabs/\*.py**

每个文件定义一个类，负责创建其对应选项卡内的所有UI组件（输入框、按钮等），并为按钮绑定命令。命令函数负责从UI获取输入，调用saes\_interface中对应的方法，最后将结果显示在输出框中。

**core/saes\_core.py**

实现了S-AES算法的所有底层原语，如nibble\_substitution(半字节代替),shift\_rows(行移位),mix\_columns(列混淆),key\_expansion(密钥扩展)等。encrypt和decrypt函数严格按照S-AES标准流程调用这些原语。

**core/saes\_interface.py**

GUI调用的唯一入口。SAESGUI实例化此类。它实现了如encrypt\_ascii(处理字符串和分组填充)、double\_encrypt、meet\_in\_middle\_attack、cbc\_encrypt(实现CBC链式逻辑)等所有作业要求的上层功能。

**4.扩展与维护**

**修改UI:**所有UI代码都集中在gui/tabs/中。修改布局或组件不会影响后端加密逻辑。

**替换算法:**若要将S-AES替换为其他16位分组算法，只需重写core/saes\_core.py中的函数，只要saes\_interface.py调用的encrypt和decrypt函数签名不变，上层功能（如CBC、多重加密）几乎无需改动。

**增加新功能:**可以遵循现有模式，例如：

1. 在core/saes\_interface.py中添加实现新功能的函数。
2. 在gui/tabs/目录中创建一个新的new\_feature\_tab.py。
3. 在gui/main\_window.py的create\_tabs方法中加载这个新选项卡。