本课题主要包括：将需要隐写的数据寄生至linux的可执行程序，从可执行程序中提取出隐写的数据， 分析linux二进制程序和评估该算法的隐蔽性。

系统大概流程：编写ELF解析器解析ELF文件格式，分析出其数据段和节的位置，找到合适的地方插入需要隐写的数据，在内存里面重新构造新的二进制文件，最后覆盖原来的可执行程序。

该程序解决某些隐写术不够隐蔽的问题。传统的隐写术将数据隐写进入图像或者音频中，安全人员可以比较轻松地提取出数据， 或者图像或音频丢失导致隐写数据丢失等隐患。 本技术基于silvio病毒感染算法，可以轻松将隐写数据寄生至各个可执行程序， 相比一般隐写术安全，也不用担心载体丢失导致隐写数据的丢失。

该系统主要完成以下几个功能模块:

1. 头文件: 整个程序的蓝图，对整个程序的函数和数据结构进行声明和定义，叙述程序功能。(预计500行)
2. 通用函数: 一些通用函数， 包括但不限异常错误处理函数， 获取文件信息函数， 分析需要隐写的数据的函数等。(500-1000行左右)
3. 主函数: main函数，用于控制整个程序，是整个程序的入口。(300行左右)
4. 信息源模块: 需要隐写的数据在这里处理。（200行左右)
5. 信息寄生模块: 这是整个程序最核心的部分，负责分析ELF文件格式以及处理ELF文件格式，找到合适的地方进行寄生，达到隐写数据的目的。(700行左右）
6. 信息提取模块: 从可执行程序提取隐写的数据。（300行左右）
7. 隐写数据预处理模块: 对隐写信息进行适当的加密，可能采用aes，des或者xor等加密方式对数据进行更进一步的保护。(500行左右)
8. 系统功能测试模块: 编写bash脚本对系统的正确性进行测试，保证系统能正常工作。