《数据安全》实验报告

姓名： 费泽锟 学号： 2014074 班级： 信安班

**实验名称：**

零知识证明实践

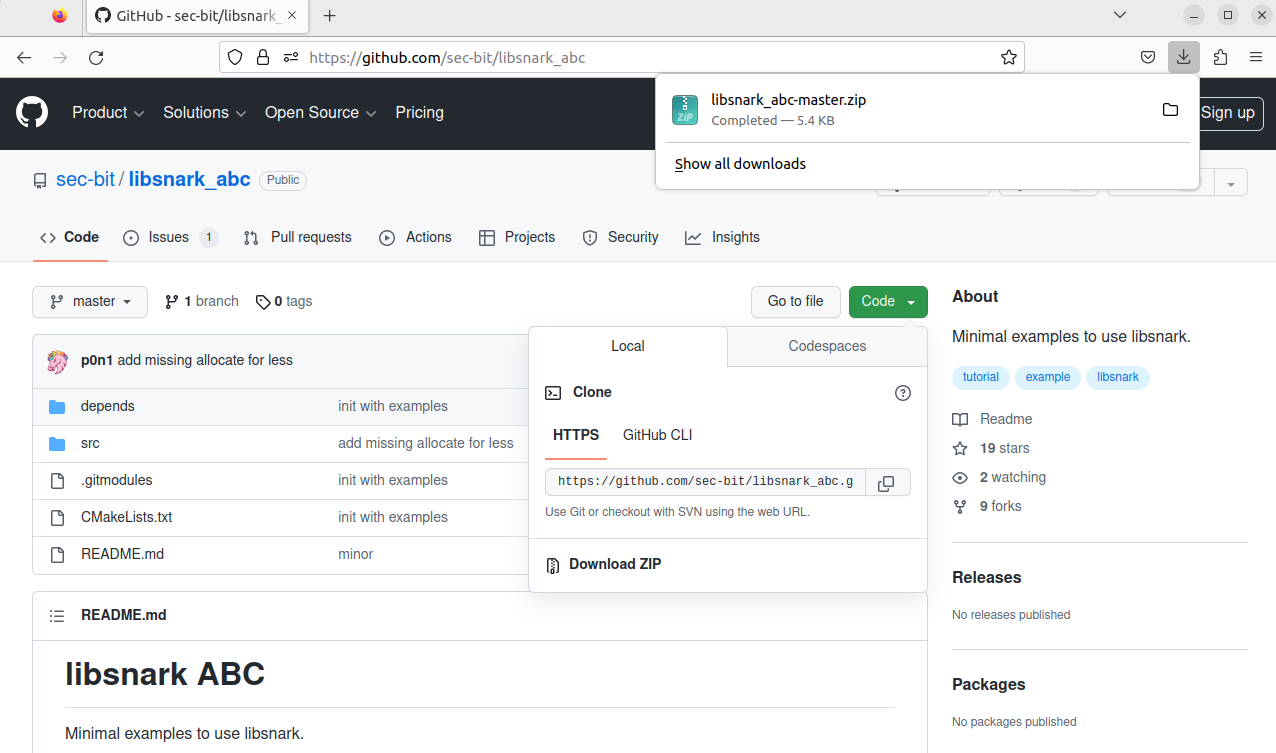
**实验要求：**

参考教材实验3.1，假设Alice希望证明自己知道如下方程的解，其中out是大家都知道的一个数，这里假设out为35，而就是方程的解，请实现代码完成证明生成和证明的验证。

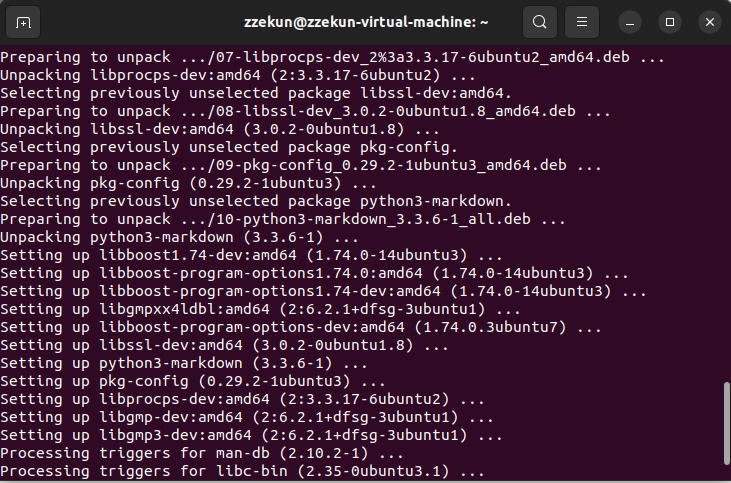
**实验过程：**

1. 配置实验环境

在进行实验之前，我们首先需要对实验环境进行配置，首先我们考虑的是能否在Ubuntu22.04虚拟机上部署libsnark实验环境，我们现在Ubuntu22.04虚拟机环境之下进行libsnark部署尝试，首先是下载好对应的压缩包：

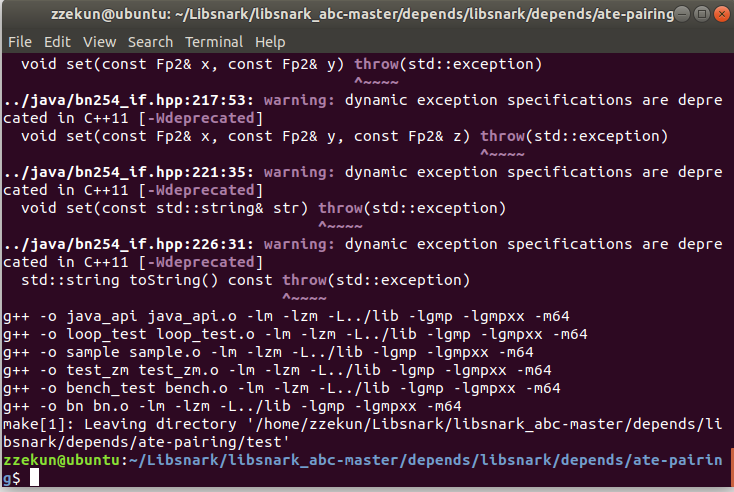


但是当我们进行depends中的子模块安装时，当我们进行子模块 ate-pairing的安装时，在运行了make -j的命令后，会发现出现了C++编译错误。这个错误的主要原因是因为C++ 17版本之中，取消了try catch语法的使用，所以会出现的错误。

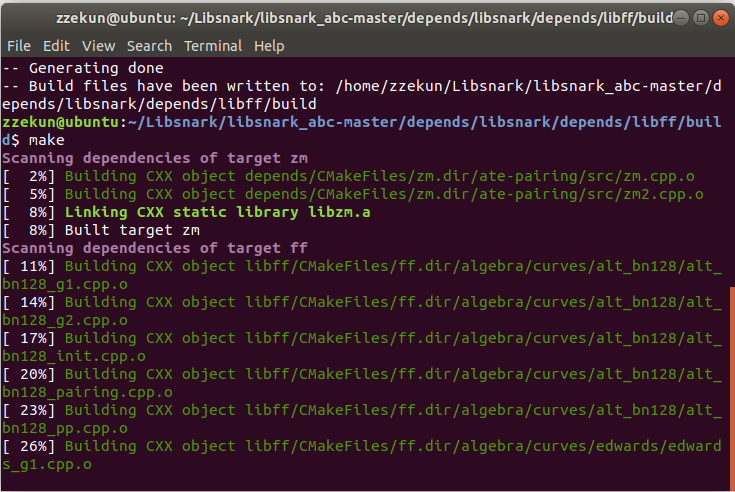


为了解决这个问题，有三种解决方案，一种为设置编译选项，一种为修改源码，最后一种方法为使用Ubuntu18.04的环境。如果修改了源码，有可能出现源码依赖而出现更多的bug，所以这里选择使用Ubuntu18.04的虚拟机环境。

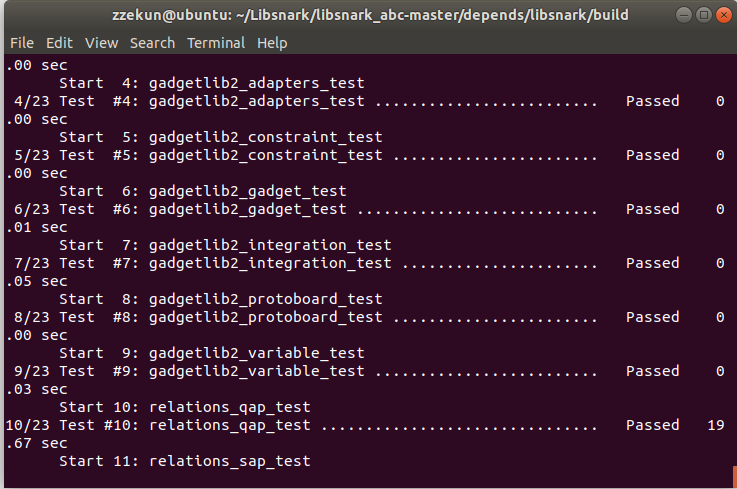
在虚拟机Ubuntu18.04的环境之中我们能够成功安装ate-pairing子模块，安装成功显示如下：



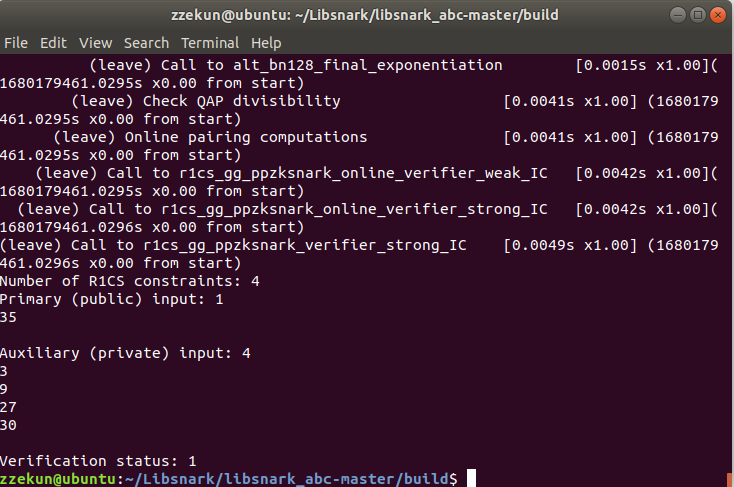
在后续的子模块安装的步骤之中，需要特别注意的一点是例如libfqfft子模块都是需要进行嵌套的子模块内容复制的，在嵌套放入两个子模块内容之后就能够成功地进行安装了。



在完成了所有的子模块的安装之后，就可以对libsnark进行编译安装了，按照实验指导书的内容，实现libsnark的编译安装。



最后通过测试/src/test的样例测试实验环境是否安装成功。测试成功后的样例结果图：

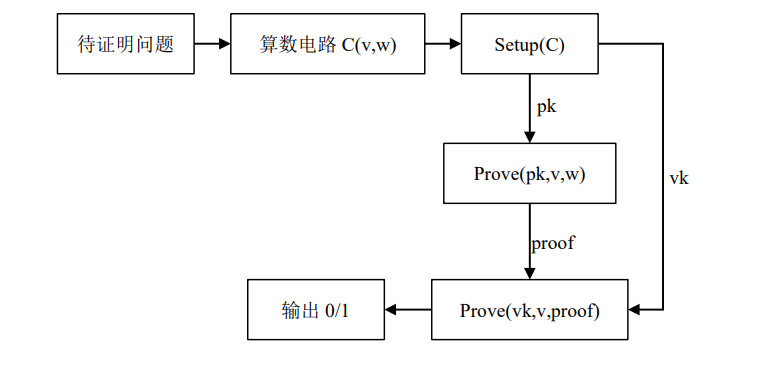


1. 参考实验3.1的复现

根据实验3.1的样例对于电路板protoboard进行学习，零知识证明的实现过程如下：

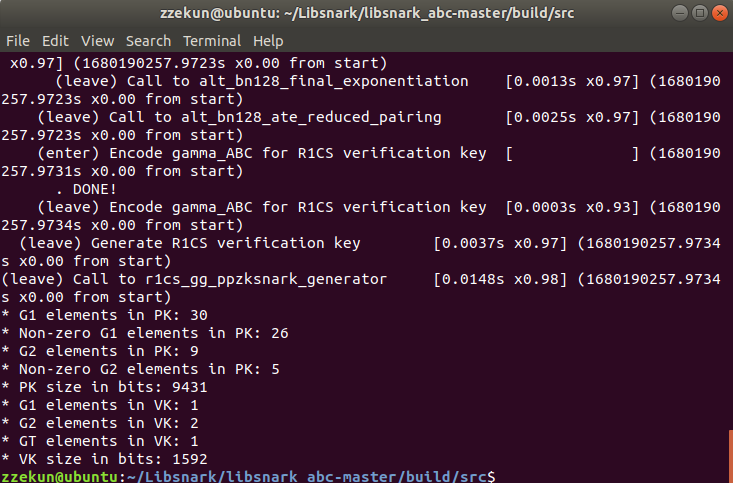
1. 将待证明的命题表达为 R1CS
2. 用 R1CS 描述电路
3. 使用原型板搭建电路
4. 生成证明密钥和验证密钥
5. 证明方使用证明密钥和其可行解构造证明
6. 验证方使用验证密钥验证证明方发过来的证明

具体的零知识证明的流程图如下：



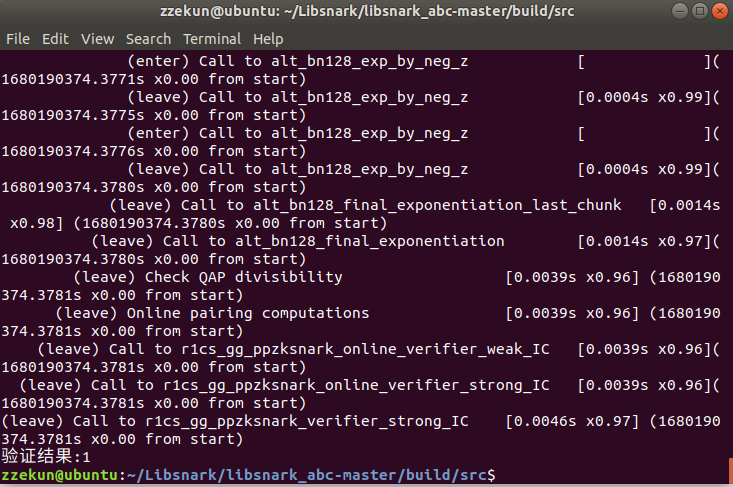
在实验过程之中，我们直接在common.hpp的共有文件之中设置外部共有变量out为0，然后通过设置secret数组对于秘密值进行输入，通过设定约束，以及生成证明密钥和验证密钥进行秘密的验证。

我们可以运行./mysetup先生成证明密钥和验证密钥，这一部分需要先获取common.hpp之中的电路约束关系。运行如下：



我们接着只要运行./myprove，只需要输入一个秘密值，这里输入秘密值2，就会接着生成其余的秘密值，但是这里需要注意一点就是common.hpp只定义了电路的约束关系，但是无法没办法实际上实现秘密值的顺序获取，只能在myprove过程之中运算得出其余的秘密值输入。

接着运行./myverify后，可以得到结果如下，验证结果为1则为验证成功。

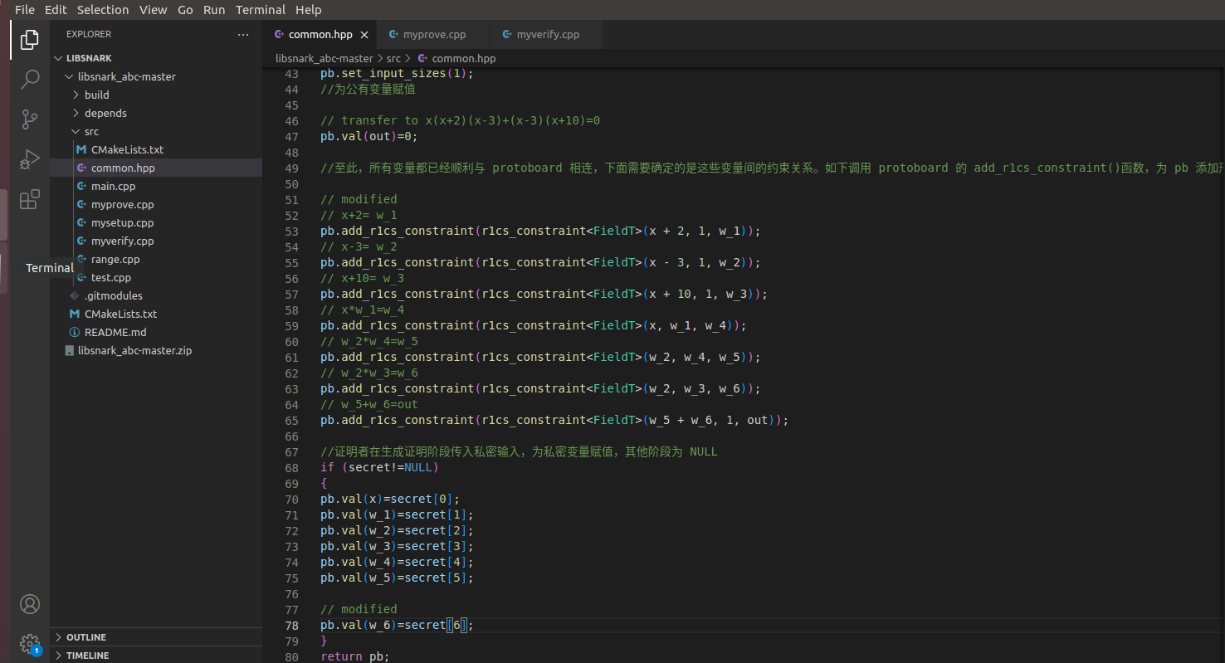


1. 解验证的实现

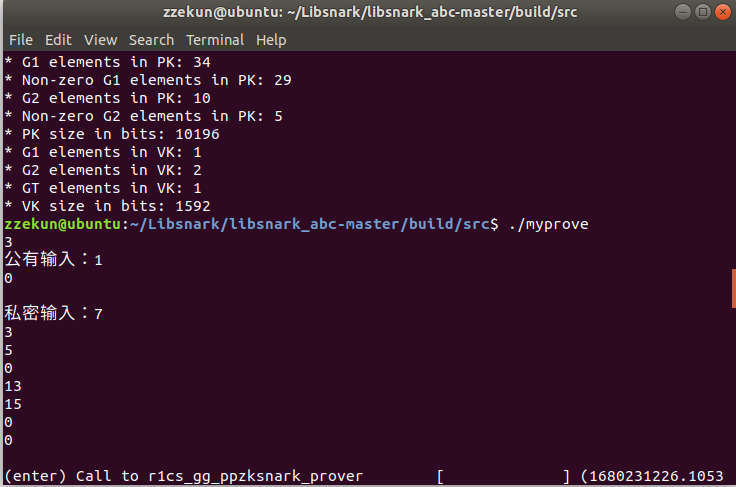
因为这里默认out的数据值为35，则可以将验证的方程转化为：，转化为上述形式之后就可以进行因式分解，转化为加法与乘法组合的形式，转化后的结果为：

接着我们就可以将电路拍平得到的函数如下：

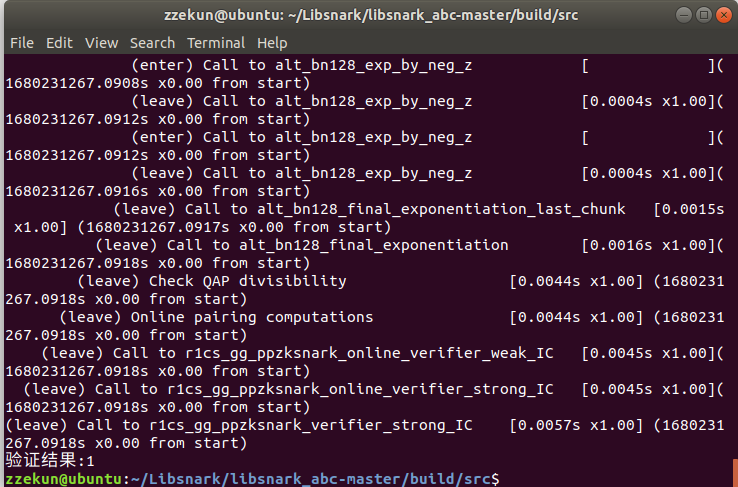
我们就可以根据实验3.1给出的范例来改写约束条件，实现验证代码，实现的代码部分如下：

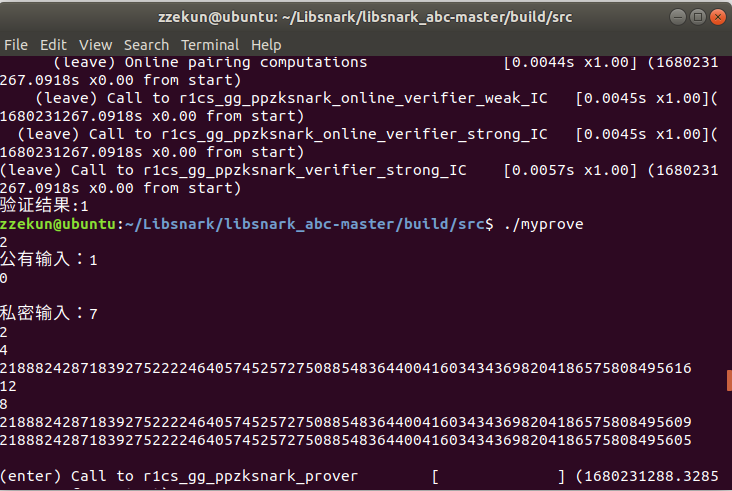


接着我们就可以使用对应的cmake命令进行编译链接得到可执行程序，接着我们可以运行./myprove接着输入正确的解3，那么期待的对应的验证结果应当为1，也就是验证通过。

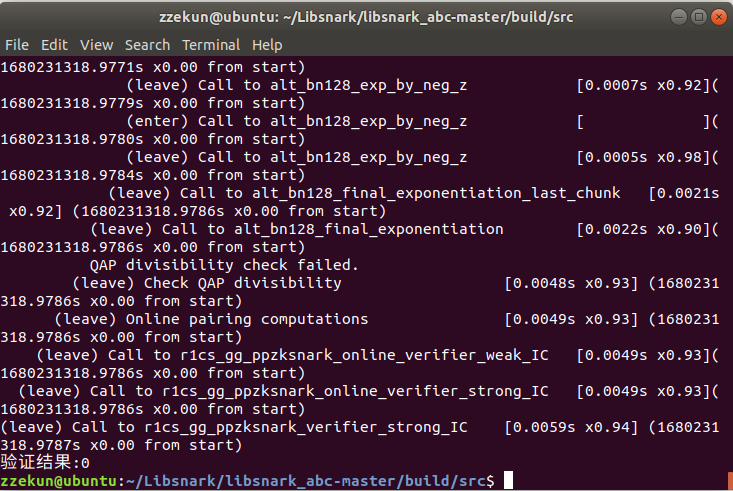


接着运行./myverift之后，就可以发现验证成功。



如果秘密输入的数据值为2，那么应当对应的实验验证结果为0，也就是验证失败。 

对应的验证结果为0



**心得体会**

通过本次实验，我们学习了零知识证明的原理，实验中给出了一个非常贴近现实的使用libsnark的场景，我们完成了最终证明生成，证明验证的过程。

经过学习与实践，将在密码学课上学习的知识和数据安全学习的内容结合在了一起，更好的理解了零知识证明和zkSNARK的特性及其适合的应用场景。