无 root 权限安装OpenFHE教程

前言

由于本人的实验都是在实验室的服务器上进行，且和同态机密相关，因此需要在服务器上安装OpenFHE。然而服务器仅仅是给我分配了一个用户空间，并没有root权限，因此需要绕开 sudo 等指令去安装。在网上找了很久都没有找到合适的安装教程，最后经过本人的一些探索成功安装，特此分享安装教程。

安装步骤

前期准备

- 确认安装路径

```
mkdir $HOME/openfhe_env
```

- 设置环境变量并永久保存

```
echo 'export PREFIX=$HOME/openfhe_env' >> ~/.bashrc
echo 'export PATH=$PREFIX/bin:$PATH' >> ~/.bashrc
echo 'export LD_LIBRARY_PATH=$PREFIX/lib:$LD_LIBRARY_PATH' >> ~/.bashrc
mkdir -p $PREFIX
```

安装相关库

- 安装 GMP（版本自由选择）

```
cd ~
wget https://gmplib.org/download/gmp/gmp-6.3.0.tar.xz
tar -xf gmp-6.3.0.tar.xz
cd gmp-6.3.0
./configure --prefix=$PREFIX --enable-cxx
make -j$(nproc)
make install
```

- 安装 NTL（版本自由选择）

```
cd ~
wget https://libntl.org/ntl-11.5.1.tar.gz
tar -xzf ntl-11.5.1.tar.gz
cd ntl-11.5.1/src
./configure PREFIX=$PREFIX \
GMP_PREFIX=$PREFIX \
CXXFLAGS="-O2 -fPIC" \
NTL_THREADS=on
make -j$(nproc)
make install
```

正式安装OpenFHE

- 安装OpenFHE

```
cd ~
git clone https://github.com/openfheorg/openfhe-development.git
cd openfhe-development
mkdir build && cd build

cmake .. \
-DCMAKE_INSTALL_PREFIX=$PREFIX \
-DCMAKE_BUILD_TYPE=Release \
-DWITH_BE2=ON \
-DWITH_BE4=ON \
-DWITH_EXAMPLES=ON \
-DWITH_TESTS=ON \
-DGMP_DIR=$PREFIX \
-DNTL_DIR=$PREFIX \
-DCMAKE_PREFIX_PATH=$PREFIX

make -j$(nproc)
make install
```

验证OpenFHE是否安装成功

编写测试程序 demo.cpp

```
#include <openfhe.h>
#include <iostream>

using namespace lbcrypto;

int main() {
    // 设置 CKKS 参数
    uint32_t multDepth = 3;
    uint32_t scaleModSize = 50;
    uint32_t batchSize = 8;

    CCParams<CryptoContextCKKSRNS> parameters;
    parameters.SetMultiplicativeDepth(multDepth);
    parameters.SetScalingModSize(scaleModSize);
    parameters.SetBatchSize(batchSize);

    // 创建加密上下文
    CryptoContext<DCRTPoly> cc = GenCryptoContext(parameters);
    cc->Enable(PKE);
    cc->Enable(LEVELEDSHE);
    cc->Enable(ADVANCEDSHE);

    std::cout << "CKKS scheme is using ring dimension: " << cc->GetRingDimension() <<
std::endl;

    // 生成密钥对
    auto keys = cc->KeyGen();
    cc->EvalMultKeyGen(keys.secretKey);

    // 创建测试数据
    std::vector<double> input = {1.0, 2.0, 3.0, 4.0, 5.0, 6.0, 7.0, 8.0};
    Plaintext ptxt = cc->MakeCKKSPackedPlaintext(input);

    // 加密
    auto ciph = cc->Encrypt(keys.publicKey, ptxt);
    std::cout << "Encryption completed." << std::endl;

    // 同态操作：加法和乘法
    auto ciphAdd = cc->EvalAdd(ciph, ciph); // ciph + ciph
    auto ciphMult = cc->EvalMult(ciph, ciph); // ciph * ciph

    // 解密
    Plaintext resultAdd, resultMult;
    cc->Decrypt(keys.secretKey, ciphAdd, &resultAdd);
    cc->Decrypt(keys.secretKey, ciphMult, &resultMult);

    // 输出结果
    std::cout << "Original input: " << input << std::endl;
```

```
    std::cout << "Decrypted addition result: " << resultAdd << std::endl;
    std::cout << "Decrypted multiplication result: " << resultMult << std::endl;

    return 0;
}
```

编译测试程序

```
g++ demo.cpp -o demo_openfhe   \
-I$PREFIX/include/openfhe   -I$PREFIX/include/openfhe/pke  \
-I$PREFIX/include/openfhe/core   -I$PREFIX/include/openfhe/binfhe \
-L$PREFIX/lib   -lOPENFHEbinfhe -lOPENFHEcore -lOPENFHEpke  \
-lntl -lgmp -lm   \
-std=c++17
```

运行测试

```
./demo_openfhe
```

输出结果

```
CKKS scheme is using ring dimension: 16384
Encryption completed.
Original input: [ 1 2 3 4 5 6 7 8 ]
Decrypted addition result: (2, 4, 6, 8, 10, 12, 14, 16,  ... ); Estimated precision: 43
bits

Decrypted multiplication result: (1, 4, 9, 16, 25, 36, 49, 64,  ... ); Estimated
precision: 40 bits
```