|  |  |
| --- | --- |
| 文件编号： |  |
| **当前版本：** | **1.0** |
| **保密级别** | [ ]秘密 [ ]机密 [ ]绝密 |
| **文件状态** | [ ]草稿 [ ]修改版 [ ]正式版 |
| **文件状态** | [ ]草稿 [ ]修改版 [ ]正式版 |

**基于MLWE的通用和模块化的**

**密钥封装机制**

**测试报告**

（Linux版）

2019年2月

**版本控制信息**

| 版本 | 日期 | 拟稿和修改 | 说明 |
| --- | --- | --- | --- |
| 1.0 | 2019-02-21 | 拟稿 | 吴洪祥，巩博儒 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

目 录

[第一章 概述 1](#_Toc4235726)

[1.1 测试目的 1](#_Toc4235727)

[1.2 测试对象及范围 1](#_Toc4235728)

[1.2.1 密钥生成 1](#_Toc4235729)

[1.2.2 密钥封装 2](#_Toc4235730)

[1.2.3 密钥解封装 2](#_Toc4235731)

[1.3 术语列表 2](#_Toc4235732)

[1.4 参考资料 3](#_Toc4235733)

[第二章 测试环境及说明 4](#_Toc4235734)

[2.1 测试环境物理拓扑结构 4](#_Toc4235735)

[2.2 硬件环境 4](#_Toc4235736)

[2.3 软件环境 4](#_Toc4235737)

[2.4 测试程序说明 4](#_Toc4235738)

[2.5 测试方法 6](#_Toc4235739)

[第三章 功能测试 10](#_Toc4235740)

[3.1 AKCN-MLWE-KT-正常功能测试 10](#_Toc4235741)

[3.2 AKCN-MLWE-KT-密文篡改测试 10](#_Toc4235742)

[3.3 AKCN-MLWE-PKE-正常功能测试 11](#_Toc4235743)

[3.4 AKCN-MLWE-PKE-密文篡改测试 11](#_Toc4235744)

[第四章 性能测试 12](#_Toc4235745)

[4.1 时间性能 12](#_Toc4235746)

[4.1.1 AKCN-MLWE-KT-密钥生成 12](#_Toc4235747)

[4.1.2 AKCN-MLWE-KT-封装 13](#_Toc4235748)

[4.1.3 AKCN-MLWE-KT-解封装 14](#_Toc4235749)

[4.1.4 AKCN-MLWE-PKE-密钥生成 15](#_Toc4235750)

[4.1.5 AKCN-MLWE-PKE-封装 16](#_Toc4235751)

[4.1.6 AKCN-MLWE-PKE-解封装 17](#_Toc4235752)

[4.2 时钟周期 19](#_Toc4235753)

[4.2.1 AKCN-MLWE-KT-密钥生成 19](#_Toc4235754)

[4.2.2 AKCN-MLWE-KT-封装 20](#_Toc4235755)

[4.2.3 AKCN-MLWE-KT-解封装 21](#_Toc4235756)

[4.2.4 AKCN-MLWE-PKE-密钥生成 21](#_Toc4235757)

[4.2.5 AKCN-MLWE-PKE-封装 23](#_Toc4235758)

[4.2.6 AKCN-MLWE-PKE-解封装 23](#_Toc4235759)

[4.3 空间性能 25](#_Toc4235760)

[4.3.1 AKCN-MLWE-KT算法的空间消耗 25](#_Toc4235761)

[4.3.2 AKCN-MLWE-PKE算法的空间消耗 25](#_Toc4235762)

[第五章 测试总结 26](#_Toc4235763)

# 概述

## 测试目的

1. 基于理想格的密钥协商和密码封装机制算法的功能验证。

2. 基于理想格的密钥协商和密码封装机制算法的性能测试。

## 测试对象及范围

基于理想格的密钥协商和密码封装机制算法程序实现包含两部分，分别是：

* AKCN-MLWE-KT。
* AKCN-MLWE-PKE。

本次测试针对这四部分算法进行功能验证和性能测试，具体包括：

1. 验证算法功能；

2. 测试算法的实现性能；

3. 给出测试平台上的存储需求。

算法各部分的API接口分别为：

### 密钥生成

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

函数名 ：kem\_keygen

功能描述：基于AKCN-MLWE-KT的密钥生成

输入参数：无

输出参数：pk -生成的公共密钥

sk -生成的私有密钥

返回值 ：0-生成密钥成功

-1-内部处理错误

-2-接口参数错误

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

int kem\_keygen( unsigned char \*pk, unsigned char \*sk);

### 密钥封装

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

函数名 ：kem\_enc

功能描述：基于AKCN-MLWE-KT的密钥封装

输入参数：pk -接收方的公钥

输出参数：ss -共享密钥

ct -密文

返回值 ：0 -成功

-1-内部处理错误

-2-接口参数错误

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

int kem\_enc(unsigned char \* pk,unsigned char \* ss, unsigned char \* ct);

### 密钥解封装

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

函数名 ：kem\_dec

功能描述：基于AKCN-MLWE-KT的密钥解封装

输入参数：sk -接收方的私钥

ct -密文

输出参数：ss -生成的共享密钥

返回值 ：0 -成功

-1-内部处理错误

-2-接口参数错误

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

int kem\_dec(unsigned char \* sk, unsigned char \* ct, unsigned char \* ss);

## 1.3 术语列表

本文中涉及的专业术语见表1.1：

表1.1 术语

|  |  |
| --- | --- |
| CPUCYCLE | CPU周期又称[机器周期](https://baike.baidu.com/item/%E6%9C%BA%E5%99%A8%E5%91%A8%E6%9C%9F/276831)，从内存读取一条[指令字](https://baike.baidu.com/item/%E6%8C%87%E4%BB%A4%E5%AD%97/3220363)的最短时间来定义，这个基准时间就是CPU周期（机器周期）。本文中除了使用时间来衡量算法性能，也给出了算法消耗的时钟周期。 |

## 1.4 参考资料

《算法文本-基于模格MLWE的通用和模块化的密钥协商和密钥封装协议》

# 测试环境及说明

## 测试环境物理拓扑结构

仅单机

## 硬件环境

**设备类型**：微型计算机

**型号**：HP Pavilion Gaming Desktop PC

**CPU**：Intel® Core™ i7-8700 CPU @ 3.20GHz \*12

**内存**：RAM 8GB(1\*8GB) DDR4 2666 NECC

**硬盘**：HDD 1TB 7200RPM SATA 3.5 cDT 2nd

SSD 128G 2280 PCIe NVMe Value

**显卡**：GFX Nvd GeF GTX1060 6GB

## 软件环境

**操作系统**：Ubuntu 18.04.1 LTS

**开发语言**：C

**编译工具**：gcc version 7.3.0 (Ubuntu 7.3.0-27ubuntu1~18.04)

## 测试程序说明

测试程序的发布方式参照开源库通用方式，即在算法发布库中集成对应的测试验证程序。整个算法库的源码组织结构分为四部分，分别是：

* AKCN-MLWE-KT子目录，包含了AKCN-MLWE-KT-Recommended算法实现及其测试程序；
* AKCN-MLWE-PKE子目录，包含了AKCN-MLWE-PKE-Recommended算法实现及其测试程序；

因算法库依赖开源库openssl，在进行算法库和测试程序的编译前请先编译安装openssl。先从官网下载源码包，然后执行如下命令编译安装：

tar -zxvf openssl-1.1.0j.tar.gz

cd openssl-1.1.0j

sudo ./config

sudo make

sudo make install

测试程序含功能的验证和性能测试。测试程序与算法库一样均为c语言开发，具体编译方法如下(linux,gcc)：

获取算法源码tar包后执行如下命令即可：

tar -xvf mlwe-kem-linux.tar

./makeall.sh

编译生成以下四个可执行程序

./AKCN-MLWE-PKE/kem\_test

./AKCN-MLWE-KT/kem\_test

所有测试程序均采用列表菜单方式提供不同项目的测试，使用时执行对应算法的测试程序即可进入测试菜单，菜单内容包括功能测试部分和交互式性能测试部分，图2.1为AKCN-MLWE-KT算法测试验证可执行程序kem\_test的测试界面：

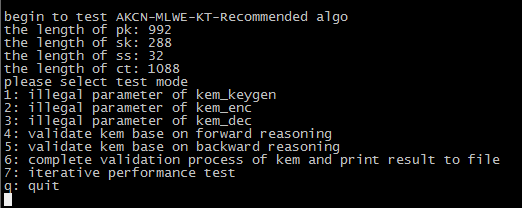


图2.1 测试界面

选择对应菜单项并输入编号即可开始测试。

## 测试方法

1. 测试案例的组织分三个层次：a.算法库实现的健壮性、稳定性验证；b.算法功能验证；c.算法性能测试。
2. 测试案例中对成功和失败的判断均指测试程序按指定逻辑返回特定的错误码。
3. 为获取性能测试准确数据，测试将对所有接口分别进行不同数量级的重复测试以获取统计数据的收敛值。

4、各算法总体测试流程见图2.2。



图2.2 总体测试流程图

基于理想格的密钥协商和密码封装机制算法的通用测试流程（含密文篡改测试）见图2.3。



图2.3 基于理想格的密钥协商和密码封装机制算法通用测试流程图

性能测试流程即在通用测试流程基础上对算法库各API接口的时间和CPU周期消耗进行统计分析和比较。

# 功能测试

## AKCN-MLWE-KT-正常功能测试

**案例编号：okcn-kem**-p001

**案例说明：**验证AKCN-MLWE-KT能否实现发送方和接收方获取相同的共享密钥，如表3.1所示。

**对应测试程序菜单项：**1

表3.1 AKCN-MLWE-KT-正常功能测试

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **功能点** | **前提条件** | **操作流程** | 预期结果 | **是否符合预期** |
| 验证AKCN-MLWE-KT能否实现发送方和接收方获取相同的共享密钥 | 无 | 1. 调用算法库密钥生成接口，生成接收方的公钥和私钥。 2. 发送方调用算法库封装接口，使用接收方的公钥生成密文和共享密钥。 3. 接收方调用算法库解封装接口，使用自己的私钥和密文生成共享密钥。 4. 确认发送方和接收方的共享密钥是否一致，如一致则继续，否则返回验证失败。 | 发送方和接收方的共享密钥一致 | 是 |

**测试结论:** 测试通过

## AKCN-MLWE-KT-密文篡改测试

**案例编号：okcn-kem**-p002

**案例说明：**确认AKCN-MLWE-KT修改密文ct后接收方解封装是否返回错误，如表3.2所示。

**对应测试程序菜单项：**3

表3.2 AKCN-MLWE-KT-密文篡改测试

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **功能点** | **前提条件** | **操作流程** | 预期结果 | **是否符合预期** |
| 确认AKCN-MLWE-KT修改密文ct后接收方解封装是否返回错误 | 无 | 1. 调用算法库密钥生成接口，生成发送方和接收方的公钥和私钥。 2. 发送方调用算法库封装接口，使用接收方的公钥生成密文和共享密钥。 3. 篡改密文。 4. 接收方调用算法库解封装接口，使用自己的私钥和篡改后的密文来生成共享密钥， 5. 比较双方的共享密钥，如不等则继续，否则返回验证失败。 | 解封装时返回错误 | 是 |

**测试结论:** 测试通过

## AKCN-MLWE-PKE-正常功能测试

**案例编号：akcn-kem**-p001

**案例说明：**验证AKCN-MLWE-PKE能否实现发送方和接收方获取相同的共享密钥，如表3.3所示。

**对应测试程序菜单项：**1

表3.3 AKCN-MLWE-PKE-正常功能测试

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **功能点** | **前提条件** | **操作流程** | 预期结果 | **是否符合预期** |
| 验证AKCN-MLWE-PKE能否实现发送方和接收方获取相同的共享密钥 | 无 | 1. 调用算法库密钥生成接口，生成接收方的公钥和私钥。 2. 发送方调用算法库封装接口，使用接收方的公钥生成密文和共享密钥。 3. 接收方调用算法库解封装接口，使用自己的私钥和密文生成共享密钥。 4. 确认发送方和接收方的共享密钥是否一致，如一致则继续，否则返回验证失败。 | 发送方和接收方的共享密钥一致 | 是 |

**测试结论:** 测试通过

## AKCN-MLWE-PKE-密文篡改测试

**案例编号：akcn-kem**-p002

**案例说明：**确认AKCN-MLWE-PKE修改密文ct后接收方解封装是否返回错误，如表3.4所示。

**对应测试程序菜单项：**3

表3.4 AKCN-MLWE-PKE-密文篡改测试

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **功能点** | **前提条件** | **操作流程** | 预期结果 | **是否符合预期** |
| 确认AKCN-MLWE-PKE修改密文ct后接收方解封装是否返回错误 | 无 | 1. 调用算法库密钥生成接口，生成发送方和接收方的公钥和私钥。 2. 发送方调用算法库封装接口，使用接收方的公钥生成密文和共享密钥。 3. 篡改密文。 4. 接收方调用算法库解封装接口，使用自己的私钥和篡改后的密文来生成共享密钥， 5. 比较双方的共享密钥，如不等则继续，否则返回验证失败。 | 解封装时返回错误 | 是 |

**测试结论:** 测试通过

# 性能测试

性能测试包括时间性能、CPU时钟周期数据的测试及算法所需存储空间的分析。测试程序菜单包含了交互式性能测试选项，可修改脚本run\_benchmark.sh并启动算法的自动性能测试，此时测试数据将写入当前目录下的性能测试文件，自动性能测试的使用说明见表4.1。

表4.1 自动性能测试使用说明

|  |  |
| --- | --- |
| **命令** | **说明** |
| ./run\_benchmark.sh | 启动算法自动性能测试， 测试数据记录到各目录下的文件中。 |
| report.sh | 搜索run\_benchmark.sh的日志文件进行汇总 |

## 时间性能

### AKCN-MLWE-KT-密钥生成

案例编号：okcn-kem-pt01

案例说明：对AKCN-MLWE-KT算法生成密钥的性能统计。

统计指标：耗时。

测试步骤：

**第一步：**数据准备：无。

**第二步：**按一次生成的密钥数分别是1、10、100、1000、10000、100000、1000000个的情况，对生成密钥统计总耗时和单次耗时，最终归纳出单次密钥生成耗时的收敛值。相关耗时的变化情况见表4.2：

表4.2 AKCN-MLWE-KT-密钥生成耗时统计

|  |  |  |
| --- | --- | --- |
| **次数** | **总耗时(微秒)** | **平均耗时(微秒)** |
| 1 | 71 | 71 |
| 10 | 650 | 65 |
| 100 | 6493 | 64.93 |
| 1000 | 65072 | 65.07 |
| 10000 | 650642 | 65.06 |
| 100000 | 6506496 | 65.06 |
| 100000 | 6507941 | 65.08 |

AKCN-MLWE-KT算法密钥生成平均耗时统计结果如图4.1所示。

图4.1 AKCN-MLWE-KT-密钥生成平均耗时统计

**测试结论：**

在测试机上AKCN-MLWE-KT算法的单次密钥生成平均耗时约为65微秒。

### AKCN-MLWE-KT-封装

案例编号：okcn-kem-pt02

案例说明：测试AKCN-MLWE-KT算法密钥封装的性能。

统计指标：耗时。

测试步骤：

**第一步：**数据准备：每次封装前先生成一组密钥。

**第二步：**按一次进行的密钥封装数分别是1、10、100、1000、10000、100000、1000000次的情况，对密钥封装统计总耗时和单次耗时，最终归纳出单次封装耗时的收敛值。相关耗时的变化情况如表4.3所示。

表4.3 AKCN-MLWE-KT-密钥封装耗时统计

|  |  |  |
| --- | --- | --- |
| **次数** | **总耗时(微秒)** | **平均耗时(微秒)** |
| 1 | 78 | 78 |
| 10 | 840 | 84 |
| 100 | 7909 | 79.09 |
| 1000 | 77565 | 77.56 |
| 10000 | 775224 | 77.52 |
| 100000 | 7750549 | 77.51 |
| 100000 | 7751653 | 77.52 |

AKCN-MLWE-KT算法密钥封装平均耗时统计结果如图4.2所示。

图4.2 AKCN-MLWE-KT-密钥封装平均耗时统计

**测试结论：**

在测试机上AKCN-MLWE-KT算法的单次封装平均耗时约为77微秒。

### AKCN-MLWE-KT-解封装

案例编号：okcn-kem-pt03

案例说明：测试AKCN-MLWE-KT算法密钥解封装的性能。

统计指标：耗时。

测试步骤：

**第一步：**数据准备：每次解封装前先生成一组封装数据。

**第二步：**按一次进行的密钥解封装数分别是1、10、100、1000、10000、100000、1000000次的情况，对密钥解封装统计总耗时和单次耗时，最终归纳出单次解封装耗时的收敛值。相关耗时的变化情况如表4.4所示：

表4.4 AKCN-MLWE-KT-密钥解封耗时统计

|  |  |  |
| --- | --- | --- |
| **次数** | **总耗时(微秒)** | **平均耗时(微秒)** |
| 1 | 19.00 | 19.00 |
| 10 | 186.00 | 18.60 |
| 100 | 1904.00 | 19.04 |
| 1000 | 18793.00 | 18.79 |
| 10000 | 187995.00 | 18.80 |
| 100000 | 1877961.00 | 18.78 |
| 100000 | 1878058.00 | 18.78 |

AKCN-MLWE-KT算法密钥解封装平均耗时统计结果如图4.3所示。

图4.3 AKCN-MLWE-KT-密钥解封平均耗时统计

**测试结论：**

在测试机上AKCN-MLWE-KT算法的单次解封装平均耗时约为18微秒。

### AKCN-MLWE-PKE-密钥生成

案例编号：akcn-kem-pt01

案例说明：对AKCN-MLWE-PKE算法生成密钥的性能统计。

统计指标：耗时。

测试步骤：

**第一步：**数据准备：无。

**第二步：**按一次生成的密钥数分别是1、10、100、1000、10000、100000、1000000个的情况，对生成密钥统计总耗时和单次耗时，最终归纳出单次密钥生成耗时的收敛值。相关耗时的变化情况如表4.5所示：

表4.5 AKCN-MLWE-PKE-密钥生成耗时统计

|  |  |  |
| --- | --- | --- |
| **次数** | **总耗时(微秒)** | **平均耗时(微秒)** |
| 1 | 72.00 | 72.00 |
| 10 | 661.00 | 66.10 |
| 100 | 6715.00 | 67.15 |
| 1000 | 66446.00 | 66.45 |
| 10000 | 663626.00 | 66.36 |
| 100000 | 6632186.00 | 66.32 |
| 100000 | 6647853.00 | 66.48 |

AKCN-MLWE-PKE算法密钥生成平均耗时统计结果如图4.4所示。

图4.4 AKCN-MLWE-PKE-密钥生成平均耗时统计

**测试结论：**

在测试机上AKCN-MLWE-PKE算法的单次密钥生成平均耗时约为66微秒。

### AKCN-MLWE-PKE-封装

案例编号：akcn-kem-pt02

案例说明：测试AKCN-MLWE-PKE算法密钥封装的性能。

统计指标：耗时。

测试步骤：

**第一步：**数据准备：每次封装前先生成一组密钥。

**第二步：**按一次进行的密钥封装数分别是1、10、100、1000、10000、100000、1000000次的情况，对密钥封装统计总耗时和单次耗时，最终归纳出单次封装耗时的收敛值。相关耗时的变化情况如表4.6所示。

表4.6 AKCN-MLWE-PKE-密钥封装耗时统计

|  |  |  |
| --- | --- | --- |
| **次数** | **总耗时(微秒)** | **平均耗时(微秒)** |
| 1 | 79.00 | 79.00 |
| 10 | 766.00 | 76.60 |
| 100 | 7748.00 | 77.48 |
| 1000 | 76661.00 | 76.66 |
| 10000 | 765561.00 | 76.56 |
| 100000 | 7647943.00 | 76.48 |
| 100000 | 7667778.00 | 76.68 |

AKCN-MLWE-PKE算法密钥封装平均耗时统计结果如图4.5所示。

图4.5 AKCN-MLWE-PKE-密钥封装平均耗时统计

**测试结论：**

在测试机上AKCN-MLWE-PKE算法的单次封装平均耗时约为76微秒。

### AKCN-MLWE-PKE-解封装

案例编号：akcn-kem-pt03

案例说明：测试AKCN-MLWE-PKE算法密钥解封装的性能。

统计指标：耗时。

测试步骤：

**第一步：**数据准备：每次解封装前先生成一组封装数据。

**第二步：**按一次进行的密钥解封装数分别是1、10、100、1000、10000、100000、1000000次的情况，对密钥解封装统计总耗时和单次耗时，最终归纳出单次解封装耗时的收敛值。相关耗时的变化情况如表4.7所示：

表4.7 AKCN-MLWE-PKE-密钥解封耗时统计

|  |  |  |
| --- | --- | --- |
| **次数** | **总耗时(微秒)** | **平均耗时(微秒)** |
| 1 | 18.00 | 18.00 |
| 10 | 184.00 | 18.40 |
| 100 | 1851.00 | 18.51 |
| 1000 | 18524.00 | 18.52 |
| 10000 | 185042.00 | 18.50 |
| 100000 | 1844661.00 | 18.45 |
| 100000 | 1853322.00 | 18.53 |

AKCN-MLWE-PKE算法密钥解封装平均耗时统计结果如图4.6所示。

图4.6 AKCN-MLWE-PKE-密钥解封平均耗时统计

**测试结论：**

在测试机上AKCN-MLWE-PKE算法的单次解封装平均耗时约为18微秒。

## 时钟周期

### AKCN-MLWE-KT-密钥生成

案例编号：okcn-kem-pc01

案例说明：对AKCN-MLWE-KT算法生成密钥的性能统计。

统计指标：cpu周期。

测试步骤：

**第一步：**数据准备：无。

**第二步：**按一次生成的密钥数分别是1、10、100、1000、10000、100000、1000000个的情况，对生成密钥统计消耗的总的CPU周期和单次CPU周期，最终归纳出单次生成密钥CPU周期消耗的收敛值。相关消耗的变化情况如表4.14所示。

表4.8 AKCN-MLWE-KT-密钥生成消耗cpu周期统计

|  |  |  |
| --- | --- | --- |
| **次数** | **总耗费cpu周期** | **平均消耗cpu周期** |
| 1 | 219569 | 219569 |
| 10 | 2039891 | 203989 |
| 100 | 20302093 | 203020 |
| 1000 | 203276219 | 203276 |
| 10000 | 2032425583 | 203242 |
| 100000 | 20329728570 | 203297 |
| 100000 | 20365887480 | 203658 |

AKCN-MLWE-KT算法密钥生成平均消耗CPU周期统计结果如图4.7所示。

图4.13 AKCN-MLWE-KT-密钥生成平均消耗cpu周期数

**测试结论：**

在测试机上AKCN-MLWE-KT算法的单次密钥生成平均CPU周期约为203000

### AKCN-MLWE-KT-封装

案例编号：okcn-kem-pc02

案例说明：测试AKCN-MLWE-KT算法密钥封装的性能。

统计指标：cpu周期。

测试步骤：

**第一步：**数据准备：每次封装前先生成一组密钥。

**第二步：**按一次密钥封装数分别是1、10、100、1000、10000、100000、1000000个的情况，对密钥封装统计消耗的总的CPU周期和单次CPU周期，最终归纳出单次密钥封装CPU周期消耗的收敛值。相关消耗的变化情况如表4.15所示。

表4.15 AKCN-MLWE-KT-密钥封装消耗cpu周期统计

|  |  |  |
| --- | --- | --- |
| **次数** | **总耗费cpu周期** | **平均消耗cpu周期** |
| 1 | 374618 | 374618 |
| 10 | 2428477 | 242847 |
| 100 | 24887249 | 248872 |
| 1000 | 243139669 | 243139 |
| 10000 | 2424759138 | 242475 |
| 100000 | 24252626325 | 242526 |
| 100000 | 24294112685 | 242941 |

图4.8是AKCN-MLWE-KT算法密钥封装CPU周期消耗收敛统计结果。

图4.14 AKCN-MLWE-KT-封装平均消耗cpu周期统计

**测试结论：**

在测试机上AKCN-MLWE-KT算法的单次封装平均CPU周期约为242900。

### AKCN-MLWE-KT-解封装

案例编号：okcn-kem-pc03

案例说明：测试AKCN-MLWE-KT算法密钥解封装的性能。

统计指标：cpu周期。

测试步骤：

**第一步：**数据准备：每次解封装前先生成一组封装数据。

**第二步：**按一次密钥解封装数分别是1、10、100、1000、10000、100000、1000000个的情况，对密钥解封装统计消耗的总的CPU周期和单次CPU周期，最终归纳出单次密钥解封装CPU周期消耗的收敛值。相关消耗的变化情况如表4.16所示。

表4.16 AKCN-MLWE-KT-解封装消耗cpu周期统计

|  |  |  |
| --- | --- | --- |
| **次数** | **总耗费cpu周期** | **平均消耗cpu周期** |
| 1 | 147108 | 147108 |
| 10 | 589764 | 58976 |
| 100 | 5894839 | 58948 |
| 1000 | 58922808 | 58922 |
| 10000 | 588111390 | 58811 |
| 100000 | 5883360400 | 58833 |
| 100000 | 5892831759 | 58928 |

图4.9是AKCN-MLWE-KT算法密钥解封装CPU周期平均消耗统计结果。

图4.15 AKCN-MLWE-KT-解封装平均消耗cpu周期统计

**测试结论：**

在测试机上AKCN-MLWE-KT算法的单次解封装平均CPU周期约为58900。

### AKCN-MLWE-PKE-密钥生成

案例编号：akcn-kem-pc01

案例说明：对AKCN-MLWE-PKE算法生成密钥的性能统计。

统计指标：cpu周期。

测试步骤：

**第一步：**数据准备：无。

**第二步：**按一次生成的密钥数分别是1、10、100、1000、10000、100000、1000000个的情况，对生成密钥统计消耗的总的CPU周期和单次CPU周期，最终归纳出单次生成密钥CPU周期消耗的收敛值。相关消耗的变化情况如表4.17所示。

表4.17 AKCN-MLWE-PKE-密钥生成消耗cpu周期统计

|  |  |  |
| --- | --- | --- |
| **次数** | **总耗费cpu周期** | **平均消耗cpu周期** |
| 1 | 221296 | 221296 |
| 10 | 2052739 | 205273 |
| 100 | 20867676 | 208676 |
| 1000 | 205795174 | 205795 |
| 10000 | 2057800627 | 205780 |
| 100000 | 20563561340 | 205635 |
| 100000 | 20569324991 | 205693 |

AKCN-MLWE-PKE算法密钥生成平均消耗CPU周期统计结果如图4.16所示。

图4.16 AKCN-MLWE-PKE-密钥生成平均消耗cpu周期数

**测试结论：**

在测试机上AKCN-MLWE-PKE算法的单次密钥生成平均CPU周期约为205700

### AKCN-MLWE-PKE-封装

案例编号：akcn-kem-pc02

案例说明：测试AKCN-MLWE-PKE算法密钥封装的性能。

统计指标：cpu周期。

测试步骤：

**第一步：**数据准备：每次封装前先生成一组密钥。

**第二步：**按一次密钥封装数分别是1、10、100、1000、10000、100000、1000000个的情况，对密钥封装统计消耗的总的CPU周期和单次CPU周期，最终归纳出单次密钥封装CPU周期消耗的收敛值。相关消耗的变化情况如表4.18所示。

表4.18 AKCN-MLWE-PKE-密钥封装消耗cpu周期统计

|  |  |  |
| --- | --- | --- |
| **次数** | **总耗费cpu周期** | **平均消耗cpu周期** |
| 1 | 251173 | 251173 |
| 10 | 2467463 | 246746 |
| 100 | 24993631 | 249936 |
| 1000 | 247583062 | 247583 |
| 10000 | 2475041977 | 247504 |
| 100000 | 24729936345 | 247299 |
| 100000 | 24738398977 | 247383 |

图4.11是AKCN-MLWE-PKE算法密钥封装CPU周期消耗收敛统计结果。

图4.17 AKCN-MLWE-PKE-封装平均消耗cpu周期统计

**测试结论：**

在测试机上AKCN-MLWE-PKE算法的单次封装平均CPU周期约为247000

### AKCN-MLWE-PKE-解封装

案例编号：akcn-kem-pc03

案例说明：测试AKCN-MLWE-PKE算法密钥解封装的性能。

统计指标：cpu周期。

测试步骤：

**第一步：**数据准备：每次解封装前先生成一组封装数据。

**第二步：**按一次密钥解封装数分别是1、10、100、1000、10000、100000、1000000个的情况，对密钥解封装统计消耗的总的CPU周期和单次CPU周期，最终归纳出单次密钥解封装CPU周期消耗的收敛值。相关消耗的变化情况如表4.19所示。

表4.19 AKCN-MLWE-PKE-解封装消耗cpu周期统计

|  |  |  |
| --- | --- | --- |
| **次数** | **总耗费cpu周期** | **平均消耗cpu周期** |
| 1 | 59610 | 59610 |
| 10 | 586818 | 58681 |
| 100 | 5884171 | 58841 |
| 1000 | 59075988 | 59075 |
| 10000 | 589245822 | 58924 |
| 100000 | 5886134330 | 58861 |
| 100000 | 5891946782 | 58919 |

图4.12是AKCN-MLWE-PKE算法密钥解封装CPU周期平均消耗统计结果。

图4.18 AKCN-MLWE-PKE-解封装平均消耗cpu周期统计

**测试结论：**

在测试机上AKCN-MLWE-PKE算法的单次解封装平均CPU周期约为58900。

## 空间性能

下面我们将针对AKCN-MLWE-KT、AKCN-MLWE-PKE 、 两种算法实现给出空间性能分析。

### AKCN-MLWE-KT算法的空间消耗

在表4.26中我们给出了AKCN-MLWE-KT算法主要参数一览。

表4.26 AKCN-MLWE-KT算法主要参数所耗空间一览表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **名称** | **公钥 pk** | **私钥 sk** | **密文ct** | **共享秘密ss** |
| 长度(byte) | 992 | 288 | 1088 | 32 |

对于具体API接口，针对大赛要求实现的kem\_api，一次密钥封装完整过程包括密钥生成函数（kem\_keygen）， 封装函数（kem\_enc）和解封装函数（kem\_dec）。表4.27给出了这三个函数输入输出参数所需要的空间 并对各个参数进行了简要的介绍。

表4.27 AKCN-MLWE-KT算法各个接口参数空间性能

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **函数** | **输入** | **长度(byte)** | **含义** | **输出** | **长度(byte)** | **含义** |
| kem\_keygen |  | | | pk | 992 | 用户公钥 |
| sk | 288 | 用户私钥 |
| kem\_enc | pk | 992 | 接收方公钥 | ct | 1088 | 密文 |
| ss | 32 | 共享秘密 |
| kem\_dec | ct | 1088 | 密文 | ss | 32 | 共享秘密 |
| sk | 288 | 接收方私钥 |

### AKCN-MLWE-PKE算法的空间消耗

在表4.28中我们给出了AKCN-MLWE-PKE算法主要参数一览。

表4.28 AKCN-MLWE-PKE算法主要参数所耗空间一览表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **名称** | **公钥 pk** | **私钥 sk** | **密文ct** | **共享秘密ss** |
| 长度(byte) | 1088 | 288 | 1152 | 32 |

对于具体API接口，针对大赛要求实现的kem\_api，一次密钥封装完整过程包括密钥生成函数（kem\_keygen）， 封装函数（kem\_enc）和解封装函数（kem\_dec）。表4.29给出了这三个函数输入输出参数所需要的空间 并对各个参数进行了简要的介绍。

表4.29AKCN-MLWE-PKE算法各个接口参数空间性能

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **函数** | **输入** | **长度(byte)** | **含义** | **输出** | **长度(byte)** | **含义** |
| kem\_keygen |  | | | pk | 1088 | 用户公钥 |
| sk | 288 | 用户私钥 |
| kem\_enc | pk | 1088 | 接收方公钥 | ct | 1152 | 密文 |
| ss | 32 | 共享秘密 |
| kem\_dec | ct | 1152 | 密文 | ss | 32 | 共享秘密 |
| sk | 288 | 接收方私钥 |

# 测试总结

1. 算法库各API对异常参数均能正常处理。
2. 验证通过了AKCN-MLWE-KT算法在正常流程下实现双方获取共享密钥的功能点。
3. 验证通过了AKCN-MLWE-PKE算法在传输时被篡改密文后解封装应报错的功能点。
4. 各算法API实现的单次平均耗时收敛值如表5.1所示（性能测试数据详见具体案例）。

表5.1 各算法API实现的性能统计表

|  |  |  |  |
| --- | --- | --- | --- |
| **算法名** | **接口名** | **平均耗时（微秒）** | **平均CPU周期消耗** |
| AKCN-MLWE-KT | 密钥生成 | 65 | 203000 |
| 封装 | 78 | 242900 |
| 解封装 | 19 | 58900 |
| AKCN-MLWE-PKE | 密钥生成 | 66 | 205700 |
| 封装 | 77 | 247000 |
| 解封装 | 18 | 58900 |