Министерство образования Республики Беларусь

Учреждение образования «Белорусский государственный технологический университет»

Факультет информационных технологий

Кафедра информационных систем и технологий

**Операционные системы и системное программирование**

Реферат

Выполнил:

студент 3 курса 4 группы

специальности ПОИТ

Матюх А.А.

Минск 2020

1. **Восстановление информации**

При восстановлении информации могут возникать ошибки связанные с нарушением целостности передаваемых данных. Эту проблему, как одну из задач, можно решить с помощью сетевой операционной системы.

В лекции использовался циклический код для решения данной задачи на основе полинома. Т.е. использовался циклический полином, где коэффициенты у полинома, полученного таким способом, равны 0 или 1 (члены полинома с нулевыми коэффициентами просто не выписываем). Такой полином основан на том, что операции вычитания на поле таких полиномов нет – операция вычитания заменяется операцией сложения полиномов. Более того, операция сложения выполняется также специфически. Никаких переносов не существует. Если число слагаемых  с одной и той же степенью k при x четное, то результатом будет 0; если нечетное – то результатом будет .

**Определение**: Циклический полином P называется первичным, если его нельзя представить как произведение  , где  и  отличны от 1 и самого P.

Далее определяются дополнительные разряды для сообщения. Для выбора числа дополнительных разрядов будем использовать результат Боуза и Чоудхури, который состоит в следующем. Пусть n – число информационных разрядов. Пусть k – число дополнительных разрядов.

Рассмотрим действия по исправлению ошибок. Нужно разделить  на первичный полином . Если деление произойдет нацело без остатка, то ошибки нет. В противном случае сразу фиксируется факт ошибки, где остаток указывает на эту ошибку.

Полученное таким образом число снова делят на первичный полином. Если в остатке получается число, отличное от 1, то производят очередной циклический сдвиг влево и снова делят и т.д. до тех пор, пока, наконец, в остатке не получится ровно 1.

1. **Шифрование Файлов**

В программировании шифрование файлов основано на криптопровайдере. Криптопровайдер – это объект, который обеспечивает методы шифрования и дешифрования. Мы получаем его таким образом:

CryptAcquireContext(

&hProv, - ссылка на криптопровайдер

NULL, - размер контейнера

NULL, - размер провайдера

PROV\_RSA\_FULL, - тип провайдера

CRYPT\_VERIFYCONTEXT)) - флаг

Ключ определяется в адресной переменной hSessionKey:

CryptEncrypt(

hSessionKey, - дескриптор ключа

0, - дескриптор хеша

true, - конец ли шифрования

0, - флаг

(BYTE\*)string, - шифруемое сообщение

&count, - длина сообщения

strlen(string)) – размер сообщения

Расшифрование:

CryptDecrypt(hSessionKey, 0, TRUE, 0, pbBuffer, &count)

Хеширование:

CryptCreateHash(

hProv, - ссылка на криптопровайдер

CALG\_MD5, - тип шифрования

0, - ключ

0, - флаг

&hHash); - адрес дескриптора хеша

CryptHashData(

hHash, - дескриптор хеша

rgbFile, - шифруемое сообщение

cbRead, - размер сообщения

1. - флаг

Агоритм RSA основан на формуле , где e и m являются открытыми числами.

1. **Элементы программирования на языке АССЕМБЛЕР**

Ассемблер – язык для работы с ячейками памяти и регистрами (памятью процессора).

Виды регистров:

1. Регистры общего назначения (РОН).

2. Флаги.

3. Указатель команд.

4. Регистры сегментов.

Регистры общего назначения размерностью в 32 бита:

EAX, ECX, EDX, EBX.

Регистры общего назначения размерностью в 16 бит:

AX, BX, CX, DX.

Регистры общего назначения размерностью в 8 бит:

AH и AL, BH и BL и т. д.

Регистры, имеющие префикс E, являются расширенными.

Accumulator register (AX) – для арифметических операций.

Counter register (CX) – для сдвигов и циклов.

Data register (DX) – для арифметических операций и операций ввода/вывода. Base register (BX) – для указателя на данные.

Stack Pointer register (SP) – для указателя вершины стека.

Stack Base Pointer register (BP) – для индикатора основания стека.

Source Index register (SI) – для указателя отправителя (источника).

Destination Index register (DI) – для получателя.

Работать на языке C++ с языком ассемблера можно двумя способами: подключить файл инструкций с расширением .asm, либо же напрямую в C++ коде с помощью блока \_asm{ }.

Основные оманды ассеблера:

Pop – очистка стека.

Push – поместить в стек.

Mov – передача данных.

Call – вызов функций.

Sub – вычитание.

Add – сложение.

Xor – логическое и/или.

Not – инверсия битов.