Министерство образования Республики Беларусь

Учреждение образования «Белорусский государственный технологический университет»

Факультет информационных технологий

Кафедра информационных систем и технологий

**Реферат**:

«Шифрование файлов»

Выполнил:

студент 3 курса 4 группы

специальности ПОИТ

Карленок Ю.А.

Минск 2020

Для **шифрования** нужно получить криптопровайдер. Криптопровайдер – это объект, который обеспечивает методы шифрования и дешифрования. Мы получаем его таким образом:

if (!CryptAcquireContext(&hProv, NULL, NULL,

PROV\_RSA\_FULL, CRYPT\_VERIFYCONTEXT))

Здесь первый аргумент получает ссылку на криптопровайдер. Указывается, что он обеспечивает шифрование по методу RSA. Детали этого метода шифрования мы приводим в конце этой лекции. Для шифрования нужно получить ключ. Ключ генерирует система таким образом:

CryptGenKey(hProv, CALG\_RC4,

CRYPT\_EXPORTABLE, &hSessionKey);

Ключ определяется в адресной переменной hSessionKey.

Шифрование выполняется по команде

if (!CryptEncrypt(hSessionKey, 0, true, 0, (BYTE\*)string,

&count, strlen(string)))

Шифруется строка string с помощью подготовленного ключа. Количество шифруемых байтов задается переменной count.

Программа расшифровки вызывается таким образом

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

Расшифрованная строка помещается в буфер pbBuffer.

Важным механизмом шифрования является **хэширование**. Особенную роль хеширование играет в системе защиты данных компьютера. Имеются разные алгоритмы хэширования. Ниже приведен код для реализации алгоритма MD5. Сначала нужно получить криптопровайдер:

if (!CryptAcquireContext(&hProv, NULL, NULL,

PROV\_RSA\_FULL,CRYPT\_VERIFYCONTEXT))

{

dwStatus = GetLastError();

printf("CryptAcquireContext failed: %d\n", dwStatus);

CloseHandle(hFile);

return dwStatus;

}

Объект криптопровайдера есть hProv. Ниже по тексту получаем объект hHash, выполняющий хеширование:

if (!CryptCreateHash(hProv, CALG\_MD5, 0, 0, &hHash))

Собственно хеширование реализуется в методе:

if (!CryptHashData(hHash, rgbFile, cbRead, 0))

Результат хеширования помещается в файл rgbFile.

**Алгоритм RSA:**

Для генерации двух ключей: тайного и открытого - используются два больших случайных простых числа, p и q. Для максимальной большей криптостойкости нужно выбирать p и q равной длины. Рассчитывается произведение: n = pq. Этой есть один из трех компонент ключа, состоящего из чисел n, e, d

Затем случайным образом выбирается второй компонент ключа (открытый ключ или ключ зашифрования, e, такой что e и (p-1)(q-1) являются взаимно простыми числами; вспомним, что (p-1)(q-1) = φ(n) – функция Эйлера.

Наконец расширенный алгоритм Евклида используется для вычисления третьего компонента ключа: ключа расшифрования, d, такого, что выполняется условие:

ed = 1 (mod φ(n)).

Другими словами:

d(-1) = e(mod φ(n)).

Таким образом, сформирован ключ, состоящий из трех чисел, которые, в свою очередь, образуют две вышеупомянутые взаимосвязанные части: открытый (публичный) ключ, (e, n), и тайный ключ, (d, n; на самом деле, как видим, тайным здесь является лишь первое из пары чисел).

Для зашифрования/расшифрования используется ключ получателя: отправитель шифрует сообщение открытым ключом, а получатель расшифровывает шифртекст своим тайным ключом.

**Зашифрование**. Если шифруется сообщение М, состоящее из r блоков: m1, m2, …, mi, …, mr, то шифротекст С будет состоять из такого же числа (r) блоков, представляемых числами:

ci = (mi)^e mod n.

**Расшифрование**. Для расшифрования каждого зашифрованного блока производится вычисление вида:

mi = (ci)^d mod n.