Златопольский Дмитрий Михайлович
Кандидат технических наук, доцент кафедры информатики и прикладной математики Московского городского педагогического университета.

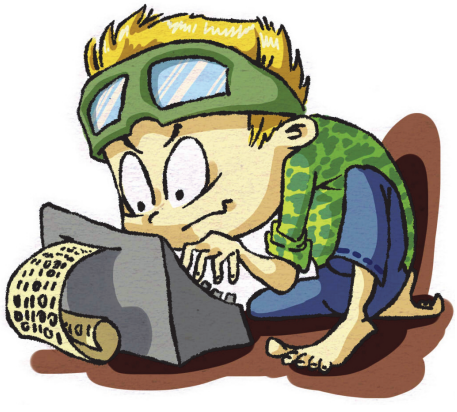
Первый урок в школе разведчиков ☺

Описывается методика шифрования и расшифровки текста с использованием логической функции **XOR** («исключающее ИЛИ»).

В статье [1] упоминалась логическая функция **XOR** (от английского *e**X**clusive **OR*** – исключающее ИЛИ). Указывалось, что в компьютере эта операция может выполняться над числами. Напомним таблицу истинности для неё в таком случае:

Первый аргумент	Второй аргумент	Результат
0	0	0
0	1	1
1	0	1
1	1	0

Если проанализировать таблицу, то можно обратить внимание на интересную деталь: если в качестве одного из аргументов взять значения результата, а в качестве второго – значения второго аргумента в таблице, то в итоге получим данные первого столбца. И то же самое со значением первого аргумента – на выходе получим значение второго аргумента. Следовательно, зная итог функции **XOR** и значение одного из аргументов, можно получить результат другого аргумента. О таком свойстве функции **XOR** говорят, что она обратима.



Из обратимости этой функции вытекает возможность её использования для шифрования текста. Возьмём текст, подлежащий шифрованию, и слово, служащее паролем (ключом) для шифрования и расшифровки. Любой текст – это последовательность символов, каждый из которых в соответствии со стандартом кодирования ASCII

представлен числом от 0 до 255. Последнее, в свою очередь, легко переводится в двоичную систему счисления, то есть текст может быть представлен как последовательность нулей и единиц (битов):

10100101110101110011010101110010101011101011100110101011100101110010011...

То же самое можно сказать и о пароле: 110101110011010101110010

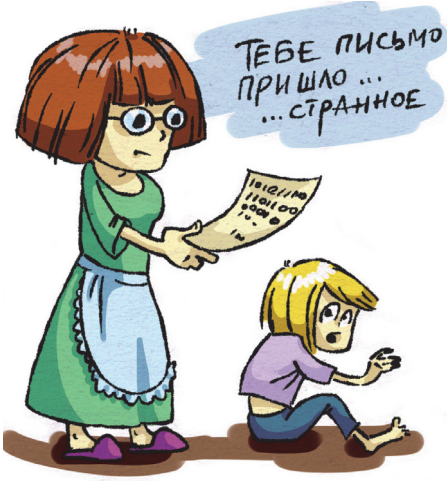
Теперь сопоставим последовательности битов шифруемого текста и пароля, повторив последний необходимое число раз. Для наглядности просто запишем их друг над другом:

10100101110101110011010101110010101011101011100110101011100101110010011...
11010111001101010111001011010111001101010111001011010111001101010111...

Далее применим функцию **XOR** к каждым двум битам, стоящим в обеих последовательностях на одинаковых местах:

01110010111000100100011111010010110011011110010110111110010100010010101...

Все – шифрование закончено!



Для расшифровки полученной последовательности к ней и к тому же самому паролю следует также применить функцию **XOR**.

Составим компьютерные программы, проводящие шифрование и расшифровку текста.

Основные этапы решения первой задачи.

1. Ввод текста, подлежащего шифрованию.
2. Ввод пароля.
3. Многократное повторение следующих действий:
   1) выделение очередного символа шифруемого текста;
   2) определение его ASCII-кода;
   3) выделение очередного символа пароля;
   4) определение его ASCII-кода;
   5) применение к значениям, полученным в пунктах 2 и 4, функции **XOR** – в результате будет получен числовой код, которым будет зашифрован очередной символ исходного текста;
   6) преобразование числового кода в строку символов;
   7) конкатенация («сцепление») этой строки символов с последовательностью, полученной для ранее зашифрованных символов.

   Количество повторений равно числу символов в шифруемом тексте.
4. Вывод результата шифрования на экран.

Остановимся на особенностях выполнения пункта 7 этапа 3. Так как код, полученный в пункте 5, может быть однозначным, двузначным или трёхзначным числом, то перед «сцеплением» его символьного представления следует приписываемое число представить в виде трёх цифр (добавить в начало числа соответствующее количество нулей). Это необходимо для того, чтобы потом, при расшифровке, можно было разбить полученную строку на 3-символьные части, соответствующие кодам отдельных символов.

Ещё одна проблема связана с разной длиной шифруемого текста и пароля. С каким символом пароля нужно сопоставлять k-й символ шифруемого текста? Можно установить, что с символом, номер которого равен $(k - 1) \bmod \text{LenPass} + 1$, где LenPass – число символов в пароле, а mod – операция определения остатка (самостоятельно убедитесь в правильности приведенной формулы!).

На языке **Бейсик** (вариант **QBasic**) программа, реализующая описанные действия, имеет вид:

```
REM Программа для шифрования текста
DIM Text$, Pass$, CodeText$, SimText$, SimPass$, SimCodeText$
DIM LenText, LenPass, AscSimText, AscSimPass, XORasc, i AS
INTEGER
CLS
INPUT "Введите текст, подлежащий шифрованию ", Text$
INPUT "Введите пароль ", Pass$
LenText = LEN(Text$)    'Длина текста
LenPass = LEN(Pass$)    'Длина пароля
CodeText$ = ""          'Зашифрованный текст
FOR i = 1 TO LenText
REM Выделяем очередной символ шифруемого текста
SimText$ = MID$(Text$, i, 1)
REM Определяем его ASCII-код
AscSimText = ASC(SimText$)
REM Выделяем очередной символ пароля
SimPass$ = MID$(Pass$, ((i - 1) MOD LenPass) + 1, 1)
REM Определяем его ASCII-код
AscSimPass = ASC(SimPass$)
REM Применяем функцию XOR
XORasc = AscSimText XOR AscSimPass
REM Преобразуем числовой код в строку символов
REM (с удалением начальных пробелов)
SimCodeText$ = LTRIM$(STR$(XORasc))
REM При необходимости добавляем
REM нули в начало числа (см. выше)
SELECT CASE VAL(SimCodeText$)
CASE 0 TO 9
SimCodeText$ = "00" + SimCodeText$
CASE 10 TO 99
SimCodeText$ = "0" + SimCodeText$
END SELECT
REM Приписываем полученное "число"
REM к имеющемуся зашифрованному тексту
```

```
CodeText$ = CodeText$ + SimCodeText$
NEXT i
REM Выводим результат на экран
PRINT CodeText$
END
```

На языке **Паскаль** соответствующая программа оформляется так:

```
{Программа для шифрования текста}
Uses CRT;
Var Text, Pass, CodeText, SimText, SimPass,  SimCodeText:
string;
LenText, LenPass, AscSimText, AscSimPass, XORasc, i: word;
BEGIN
Clrscr;
Write('Введите текст, подлежащий шифрованию ');
Readln(Text);
Write('Введите пароль ');
Readln(Pass);
LenText := Length(Text);   {Длина текста}
LenPass := Length(Pass);   {Длина пароля}
CodeText := '';          {Зашифрованный текст}
For i := 1 To LenText Do
Begin
{Выделяем очередной символ шифруемого текста}
SimText := Copy(Text, i, 1);
{Определяем его ASCII-код}
AscSimText := Ord(SimText[1]);
{Выделяем очередной символ пароля}
SimPass := Copy(Pass, (i - 1) Mod LenPass + 1, 1);
{Определяем его ASCII-код}
AscSimPass := Ord(SimPass[1]);
{Применяем функцию XOR}
XORasc := AscSimText XOR AscSimPass;
{Преобразуем число XORasc
в строку символов SimCodeText}
Str(XORasc, SimCodeText);
{При необходимости добавляем
нули в начало записи числа (см. выше)}
Case XORasc Of
0..9: SimCodeText := '00' + SimCodeText;
10..99: SimCodeText := '0' + SimCodeText
End;
{Приписываем полученное "число"
к имеющемуся зашифрованному тексту}
CodeText := CodeText + SimCodeText
end;
{Выводим результат на экран}
Writeln(CodeText)
END.
```

Вы можете усовершенствовать приведённые программы, например, добавив в них процедуру перевода полученной строки символов-цифр в строку символов, чтобы зашифрованный текст выглядел как бессмысленный набор букв.

Теперь обсудим процедуру расшифровки. Сначала следует задать строку, которую нужно расшифровать (она представляет собой последовательность символов-цифр, полученную в результате выполнения разобранной выше программы), а также пароль (ясно, что он должен быть тем же). Затем многократно повторяются следующие действия:

1) выделяются 3 очередные цифры расшифровываемой строки (они связаны с одним символом исходного текста);

2) эти 3 цифры преобразуются в число;

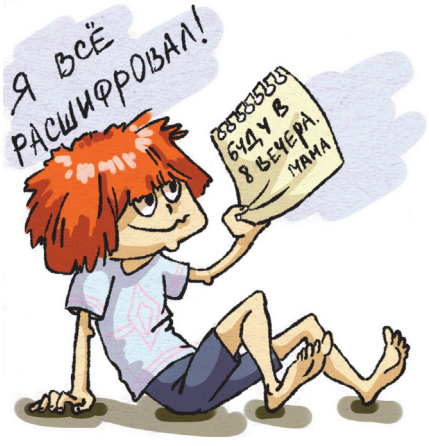
3) выделяется очередной символ пароля;

4) определяется его ASCII-код;

5) к значениям, полученным в п.п. 2) и 4), применяется функция **XOR** – в результате будет получен числовой код, которым зашифрован очередной символ исходного текста;

6) это код преобразуется в соответствующий символ, который присоединяется к последовательности ранее расшифрованных символов.

Прежде чем представлять соответствующие программы, заметим, что в них также учитываются разные длины пароля и расшифровываемого текста (см. выше).



Далее приводится текст программы на языке **Бейсик.**

```
REM Программа для расшифровки текста
DIM CodeText$, Pass$, Text$, Sim3CodeText$, SimPass$
DIM LenCodeText, LenPass, ValSim3CodeText, AscSimPass, XO
Rasc, i AS INTEGER
CLS
INPUT "Введите текст, подлежащий расшифровке ", CodeText$
INPUT "Введите пароль ", Pass$
LenCodeText = LEN(CodeText$) 'Длина текста
LenPass = LEN(Pass$)         'Длина пароля
Text$ = ""          'Расшифрованный текст
FOR i = 1 TO LenCodeText STEP 3
REM Выделяем 3 очередные цифры
REM расшифровываемой строки
Sim3CodeText$ = MID$( CodeText$, i, 3)
REM Преобразуем их в число
ValSim3CodeText = VAL(Sim3CodeText$)
REM Выделяем очередной символа пароля
SimPass$ = MID$(Pass$, (INT(i / 3) MOD LenPass) + 1, 1)
REM Определяем его ASCII-код
AscSimPass = ASC(SimPass$)
```

```
REM Применяем функцию XOR
XORasc = ValSim3CodeText XOR AscSimPass
REM Преобразуем полученный код
REM в соответствующий символ
REM и присоединяем его
REM к имеющемуся расшифрованному тексту
REM (значению величины Text$)
Text$ = Text$ + CHR$(XORasc)
NEXT i
REM Выводим результат на экран
PRINT "Расшифрованный текст "; Text$
END
```

Язык **Паскаль**

```
{Программа для шифрования текста}
Uses CRT;
Var CodeText, Pass, Sim3CodeText, SimPass, Text, SimText:
string;
LenCodeText, LenPass, AscSim3CodeText, AscSimPass, XORasc,
code, i : integer;
BEGIN
Clrscr;
Writeln('Введите текст, подлежащий расшифровке ');
Readln(CodeText);
Write('Введите пароль ');
Readln(Pass);
LenCodeText := Length(CodeText);   {Длина текста}
LenPass := Length(Pass);    {Длина пароля}
Text := '';          {Расшифрованный текст}
i := 1;
While i <= LenCodeText Do
Begin
{Выделяем 3 очередных символа зашифрованного текста}
Sim3CodeText := Copy(CodeText, i, 3);
{Преобразуем строку из трех цифр в число – ASCII-код}
Val(Sim3CodeText, AscSim3CodeText, code);
{Выделяем очередной символ пароля}
SimPass := Copy(pass, Trunc(i/3) Mod lenpass + 1, 1);
{Определяем его ASCII-код}
AscSimPass := Ord(SimPass[1]);
{Применяем функцию XOR}
XORasc := AscSim3CodeText XOR AscSimPass;
{Находим соответствующий символ}
SimText := Chr(XORasc);
{Приписываем полученный символ
к имеющемуся расшифрованному тексту
(значению величины Text)}
Text := Text + SimText;
i := i + 3
```

```
end;
{Выводим результат на экран}
Writeln(Text);
Readln
END.
```

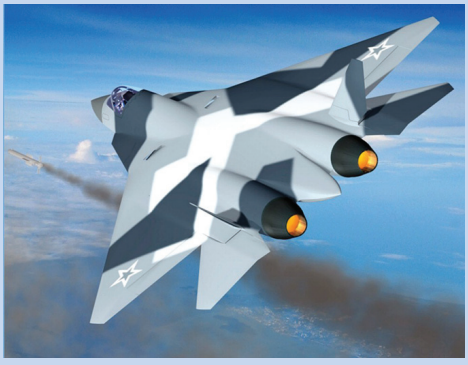
В заключение заметим, что система, при которой шифрование и расшифровка текста осуществляются с использованием одного и того же пароля (ключа), называется шифрованием с закрытым ключом, или симметричной системой шифрования.

Литература

1. *Златопольский Д.М.* Обмен без обмана // Потенциал. – 2011. – № 5.

Новости Новости Новости Новости Новости

Новый российский самолёт



В ноябре минувшего года прошло испытание третьего опытного образца нового (по внешнему виду, конструкционно и технологически) отечественного истребителя Т-50. Испытательный полёт этой перспективной (скоростной, всепогодной, невидимой для радаров и «интеллектуальной») машины, длившийся более часа, показал хорошие результаты и дал специалистам основание предполагать, что уже в следующем (2013) году Т-50 может быть принят на вооружение и первые его партии начнут поступать в военно-воздушные части российской армии.

Хотя создатели нового истребителя (сотрудники компании «Сухой») не разглашают многих сведений о его взлётно-технических характеристиках, некоторые из них известны. Т-50 может развивать скорость до 2100 км/ч, преодолевать расстояния до 5500 км. Его поверхность покрыта специальным нановеществом, снижающим видимость самолёта и сопротивление воздуха. Газотурбинный двигатель изготовлен из комплекса материалов – керамических, полимерных и металлических композитов, никелевых «жароупоров», сплавов на основе никеля и титана, что позволило повысить температуру газа в нём на 300 – 400 °С и увеличить его ресурс в 2 – 3 раза. Бортовая радиолокационная станция имеет активную фазированную антенную решётку (АФАР), входящую в состав высокоавтоматизированной многофункциональной интегрированной радиоэлектронной системы – это облегчит лётчику управление машиной и даст возможность сконцентрировать внимание на выполнении боевых тактических задач.