XI ОЛИМПИАДА ПО ИНФОРМАТИКЕ И КОМПЬЮТЕРНОЙ БЕЗОПАСНОСТИ
2016 г.
Вариант 2

Задача 1. RAR

Платежные терминалы получают индивидуальные пакеты обновлений для установленного в них программного обеспечения через сеть. При этом в целях безопасности эти пакеты пересылаются в зашифрованных архивах. Пароли шифрования для терминалов разные и администратору не известны.

Администратор на CD-R диске получил очередной незашифрованный пакет обновлений (файл *apu_test.rar*) для отладочного терминала. В ходе проверки данного пакета антивирусом оказалось, что ни один из файлов не содержит вредоносного кода.

Вместе с тем стало известно, что злоумышленники запланировали атаку на один из терминалов и, возможно, подменили некоторые файлы в пакете обновлений. С помощью специальной программы администратору удалось получить некоторые фрагменты пакета обновлений терминалов.

Проанализируйте эти фрагменты и выясните, какие из файлов в пакете были подменены.

Содержимое архива пакета обновлений с диска администратора:

Имя	Размер	CRC32
..		
apu0006.dat	1 735	B483DBD9
apu0007.dat	826	65252FF0
apu0008.dat	151	FBD03B2F
apu0009.dat	1 282	A5D76BE3
apu0010.dat	2 033	E46C5222

К задаче прилагается: исходный архив без вирусов (apu_test.rar), фрагменты пакета обновлений (apu_termX.NNN), программа-архиватор WinRAR.

Задача 2. Гамма

В центр обработки информации поступило четыре файла, каждый из которых является зашифрованным представлением изображения формата PNG. Известно, что шифрование осуществлялось методом «двоичного гаммирования», т.е. путем выполнения операции «побитового исключающего ИЛИ» между байтами исходного файла и байтами, полученными циклическим повторением последовательности из 4-х байтов ключа. Сотрудники центра успели расшифровать только три файла с именами *First.png*, *Second.png* и *Third.png*.

Помогите расшифровать оставшийся файл «*Secret.enc*». В ответе укажите слово, изображенное на полученной картинке формата PNG.

К задаче прилагается: файлы с расшифрованными картинками (First.png, Second.png, Third.png), файл с зашифрованной картинкой Secret.enc, редактор файлов в 16-ном формате (HexEditor).

Задача 3. Маршрутизация

В студенческом городке развернуто 10 локальных вычислительных сетей (ЛВС). В каждой сети есть один маршрутизатор, его номер соответствует номеру сети. Линии связи между маршрутизаторами указаны на рисунке. Соединение с Интернет имеют только маршрутизаторы с номерами 1, 2 и 5.

В служебной части сетевых пакетов имеется счетчик *S*, который увеличивается на 1 при каждой пересылке между маршрутизаторами. Из Интернет пакеты попадают в сети со счетчиком $S = 1$.

При поступлении пакета в очередной маршрутизатор с номером *R* осуществляется анализ его адреса назначения. Если сетевой пакет не предназначен какому-либо узлу из сети маршрутизатора, то он отправляется одному из соседних маршрутизаторов по правилу:

- если $S / R < 2$, то соседу с минимальным номером;
- если $S / R == 2$, то соседу со средним значением номера;
- если $S / R > 2$, то соседу с максимальным номером.

В какую сеть надо отправить пакет из Интернет, чтобы он дошел до сети с номером 10 за минимальное число шагов? Найдите это число шагов.

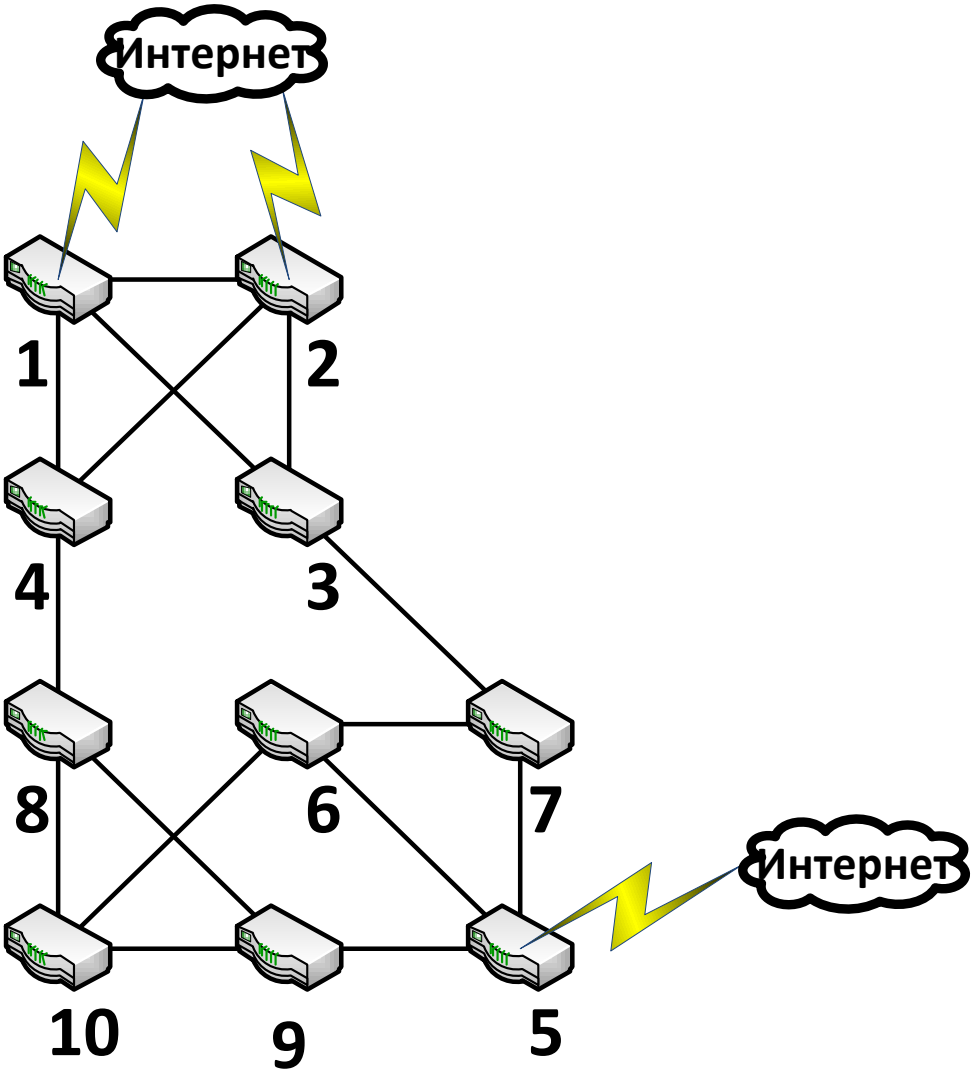

Рисунок. Схема соединения ЛВС

Задача 4. Дешифрование

Вася написал свою программу для шифрования сообщений и зашифровал с ее помощью сообщение для своего друга Миши. Чтобы усложнить понимание программы, Вася запутал код и добавил использование пароля для осуществления операций шифрования и расшифрования.

Вася отправил Мише сообщение:

```
/f-()1f?)34f1/ /f6'551)4"
```

и программу, а пароль отправить забыл. Однако Миша легко расшифровал сообщение без пароля.

Помогите Мише расшифровать сообщение и найти ключ.

Листинг программы приведен ниже.

Паскаль

```
Uses crt;
Var a, c, d, e: string; b: integer;
Begin
  Clrscr; c:= '';
  Writeln('Введите текст'); Readln(a);
  Repeat
    Writeln('Введите ключ'); Readln(d);
  Until length(d)+4>=8;
  Repeat
    b:=length(d);
    If length(e)<length(a) then
      e:=e+d[b]; b:=b+1;
  Until length(e)>=length(a);
  For b:=1 to length(a) do
    c:=c+chr(byte(a[b]) xor byte(e[b]));
  For b:=1 to length(a) do write(c[b]);
  Writeln;
  Readln;
End.
```

Си

```
#include <stdio.h>
#include <string.h>
void main()
{
   char a[100]={0}, c[100]={0},
        d[100]={0}, e[100]={0};
   int b;
   printf("Введите текст: ");
   scanf("%s", a);
   do
   {
      printf("Введите ключ: ");
      scanf_s("%s", d, 100);
   }
   while(strlen(d)+4<8);
   do
   {
      b = strlen(d)-1;
      if (strlen(e)<strlen(a))
        e[strlen(e)] = d[b]; b++;
   }
   while(strlen(e) < strlen(a));
   for (b = 0; b < strlen(a); b++)
      c[b] = a[b] ^ e[b];
   for(b = 0; b < strlen(a); b++)
      printf("%c", c[b]);
   printf("\n");
}
```

Задача 5. Защитный блок

Промышленная установка управляется по 4-разрядной шине данных. Команды по ней передаются последовательно. Для удобства записи будем интерпретировать их как символы в алфавите 0,1,2,..,9,A,B,C,D,E,F.

Известно, что некоторые цепочки команд приводят к поломке установки. Поэтому на шине планируется установить защитный блок, исправляющий такие цепочки на безопасные. Логика работы защитного блока определяется двумя таблицами. Первая из них определяет следующую активную строку в зависимости от входного символа и текущей активной строки (функция переходов). Вторая таблица определяет, что появится на выходе защитного блока в зависимости от входного символа и текущей активной строки (функция выходов). В начальный момент времени активна строка с номером 0. Фрагмент кода функции работы защитного блока приведен ниже.

Паскаль	Си

type
  matrix=array[1..n,1..m] of integer;
function GetOutput(StateMas : matrix;
  OutMas : matrix;
  InSymb : integer;
  var CurState:integer):
  integer;
var
  NewState:integer;
  OutSymb:integer;
begin
  NewState:=StateMas[CurState][InSymb];
  OutSymb:=OutMas[CurState][InSymb];
  CurState:=NewState;
  result:=OutSymb;
end;
	int GetOutput(int **StateMas,
	  int **OutMas,
	  int InSymb,
	  int& CurState)
	  // StateMas –таблица(матрица) переходов
	  // OutMas – таблица(матрица) выходов
	  // InSymb – входной символ
	  // CurState – текущее состояние
	  (меняется в результате
	  выполнения функции)
	  // RETURN – выходной символ
	{
	  int NewState;
	  int OutSymb;
	
	  NewState = StateMas[CurState][InSymb];
	  OutSymb = OutMas[CurState][InSymb];
	  CurState = NewState;
	  return OutSymb;
	}

Настройте защитный блок таким образом, чтобы он пропускал все команды, кроме запрещенных, вместо которых на выходе должна появиться безопасная выходная последовательность (см. таблицу).

Запрещенная входная последовательность	Выходная последовательность
A012AB	A012AA

Результат выполнения задачи – файл с прошивкой защитного блока.

К задаче прилагается: программа обучения и тестирования защитного блока.