## Условия задач заключительного этапа

## Задача 1. Парольная комбинация

У администратора Ивана на рабочем компьютере стоит четырехзначный пароль, состоящий из цифр. После трех неудачных попыток ввода пароля компьютер блокируется. Известно, что сумма первых двух цифр и сумма последних двух цифр пароля равны простым числам.

Помощник шпиона пригласил Ивана в кафе на обед. На какое минимальное время необходимо задержать Ивана, чтобы шпион смог гарантированно подобрать пароль от компьютера и скопировать данные, если на ввод пароля требуется 1 секунда, блокировка компьютера осуществляется на 10 секунд, а время копирования нужных данных составляет 2 минуты?

## Решение

Сумма двух цифр может принимать значения от 0 до 18. В этих границы попадают только следующие простые числа: 2, 3, 5, 7, 11, 13, 17.

Возможные комбинации половинок пароля:
сумма равна «2»: 02, 20, 11
сумма равна «3»: 03, 30, 12, 21
сумма равна «5»: 05, 50, 14, 41, 23, 32
сумма равна «7»: 07, 70, 16, 61, 25, 52, 34, 43
сумма равна «11»: 29, 92, 38, 83, 47, 74, 56, 65
сумма равна «13»: 49, 94, 58, 85, 67, 76
сумма равна «17»: 89, 98
Всего возможно 37 комбинаций.

Поскольку нет информации об отсутствии повторов пар цифр, число комбинаций из указанных пар цифр равно

$$C = 37*37 = 1\,369\ .$$

Если разбить полученное число на комбинации по три пароля, которые можно ввести до блокировки, то получим:

$$1\,369 = 456*3 + 1.$$

То есть получается 456 блокировок.

Для ввода всех паролей потребуется 1 369 секунд, время блокировки составит 456*10 = 4560 секунд.

Общее время, необходимое на ввод пароля, составит 1369 + 4560 = 5929 секунд, что равно 1 часу 38 минутам 49 секундам + 2 минуты на копирование.

**Ответ: 1 час 40 минут 49 секунд**

---

## Задача 2. Секретное сообщение

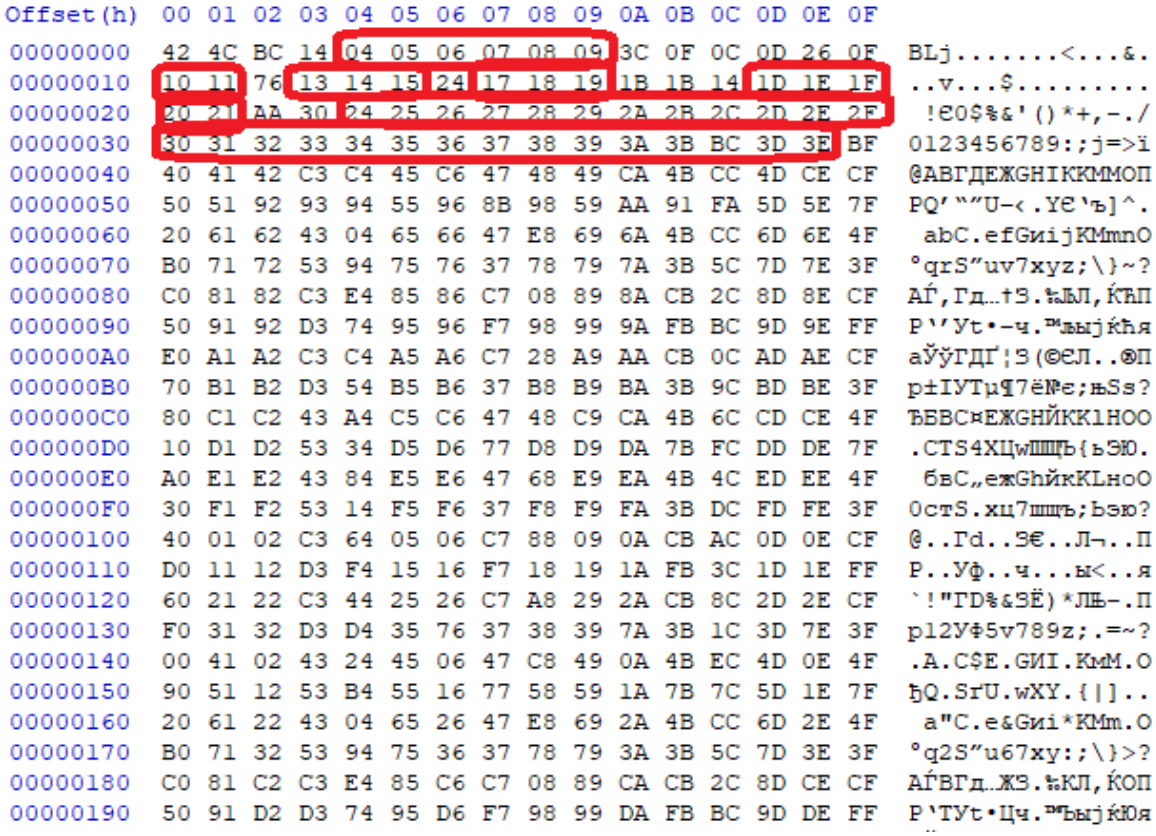
Аналитику удалось перехватить зашифрованное изображение, но программа шифрования утеряна. Известно, что шифрование осуществлялось методом «двоичного гаммирования», т.е. путем последовательного выполнения операции «побитового исключающего ИЛИ» между каждым байтом изображения и байтом ключа. Известно также, что ключ формировался в самой программе шифрования.

Восстановите текст, записанный на изображении, а также алгоритм шифрования и используемый ключ.

*К задаче прилагается:*
*изображение pic_v1.bmp.*

## Решение

Рассмотрим подробнее содержимое файла.



| Offset(h) | 00 | 01 | 02 | 03 | 04 | 05 | 06 | 07 | 08 | 09 | 0A | 0B | 0C | 0D | 0E | 0F | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 00000000 | 42 | 4C | BC | 14 | 04 | 05 | 06 | 07 | 08 | 09 | 3C | 0F | 0C | 0D | 26 | 0F | BLj.....<...&. |
| 00000010 | 10 | 11 | 76 | 13 | 14 | 15 | 24 | 17 | 18 | 19 | 1B | 1B | 14 | 1D | 1E | 1F | ..v...$..... |
| 00000020 | 20 | 21 | AA | 30 | 24 | 25 | 26 | 27 | 28 | 29 | 2A | 2B | 2C | 2D | 2E | 2F | !Є0$%&'()*+,-./ |
| 00000030 | 30 | 31 | 32 | 33 | 34 | 35 | 36 | 37 | 38 | 39 | 3A | 3B | BC | 3D | 3E | BF | 0123456789:;j=>ї |
| 00000040 | 40 | 41 | 42 | C3 | C4 | 45 | C6 | 47 | 48 | 49 | CA | 4B | CC | 4D | CE | CF | @АВГДЕЖGHIККММОП |
| 00000050 | 50 | 51 | 92 | 93 | 94 | 55 | 96 | 8B | 98 | 59 | AA | 91 | FA | 5D | 5E | 7F | PQ’“”U–‹.YЄ‘ъ]^. |
| 00000060 | 20 | 61 | 62 | 43 | 04 | 65 | 66 | 47 | E8 | 69 | 6A | 4B | CC | 6D | 6E | 4F | abC.efGиijКМmnO |
| 00000070 | B0 | 71 | 72 | 53 | 94 | 75 | 76 | 37 | 78 | 79 | 7A | 3B | 5C | 7D | 7E | 3F | °qrS”uv7xyz;\}~? |
| 00000080 | C0 | 81 | 82 | C3 | E4 | 85 | 86 | C7 | 08 | 89 | 8A | CB | 2C | 8D | 8E | CF | АЃ‚Гд…†З.‰ЉЛ,ЌЋП |
| 00000090 | 50 | 91 | 92 | D3 | 74 | 95 | 96 | F7 | 98 | 99 | 9A | FB | BC | 9D | 9E | FF | P‘’Уt•–ч.™љыjќћя |
| 000000A0 | E0 | A1 | A2 | C3 | C4 | A5 | A6 | C7 | 28 | A9 | AA | CB | 0C | AD | AE | CF | аЎўГДҐ¦З(©Є‹..®П |
| 000000B0 | 70 | B1 | B2 | D3 | 54 | B5 | B6 | 37 | B8 | B9 | BA | 3B | 9C | BD | BE | 3F | p±IУTµ¶7ё№є;њSs? |
| 000000C0 | 80 | C1 | C2 | 43 | A4 | C5 | C6 | 47 | 48 | C9 | CA | 4B | 6C | CD | CE | 4F | ЂБВС¤ЕЖGHЙККlНОО |
| 000000D0 | 10 | D1 | D2 | 53 | 34 | D5 | D6 | 77 | D8 | D9 | DA | 7B | FC | DD | DE | 7F | .СТS4ХЦwШЩЪ{ьЭЮ. |
| 000000E0 | A0 | E1 | E2 | 43 | 84 | E5 | E6 | 47 | 68 | E9 | EA | 4B | 4C | ED | EE | 4F | бвС„ежGhйкKLноО |
| 000000F0 | 30 | F1 | F2 | 53 | 14 | F5 | F6 | 37 | F8 | F9 | FA | 3B | DC | FD | FE | 3F | 0сtS.хц7шщъ;Ьэю? |
| 00000100 | 40 | 01 | 02 | C3 | 64 | 05 | 06 | C7 | 88 | 09 | 0A | CB | AC | 0D | 0E | CF | @..Гd..Зˆ..Л¬..П |
| 00000110 | D0 | 11 | 12 | D3 | F4 | 15 | 16 | F7 | 18 | 19 | 1A | FB | 3C | 1D | 1E | FF | Р..Уф..ч...ы<..я |
| 00000120 | 60 | 21 | 22 | C3 | 44 | 25 | 26 | C7 | A8 | 29 | 2A | CB | 8C | 2D | 2E | CF | `!"ГD%&ЗЁ)*ЛЊ-.П |
| 00000130 | F0 | 31 | 32 | D3 | D4 | 35 | 76 | 37 | 38 | 39 | 7A | 3B | 1C | 3D | 7E | 3F | р12У¶5v789z;.=~? |
| 00000140 | 00 | 41 | 02 | 43 | 24 | 45 | 06 | 47 | C8 | 49 | 0A | 4B | EC | 4D | 0E | 4F | .A.C$E.GИI.КмМ.О |
| 00000150 | 90 | 51 | 12 | 53 | B4 | 55 | 16 | 77 | 58 | 59 | 1A | 7B | 7C | 5D | 1E | 7F | ђQ.SґU.wXY.{\|].. |
| 00000160 | 20 | 61 | 22 | 43 | 04 | 65 | 26 | 47 | E8 | 69 | 2A | 4B | CC | 6D | 2E | 4F | a"C.e&Gиi*КМm.О |
| 00000170 | B0 | 71 | 32 | 53 | 94 | 75 | 36 | 37 | 78 | 79 | 3A | 3B | 5C | 7D | 3E | 3F | °q2S”u67xy:;\}>? |
| 00000180 | C0 | 81 | C2 | C3 | E4 | 85 | C6 | C7 | 08 | 89 | CA | CB | 2C | 8D | CE | CF | АЃВГд…ЖЗ.‰КЛ,ЌОП |
| 00000190 | 50 | 91 | D2 | D3 | 74 | 95 | D6 | F7 | 98 | 99 | DA | FB | BC | 9D | DE | FF | Р‘ТУt•Цч.™ЪыjќЮя |

Рисунок 1 – Содержимое зашифрованного файла

В содержимом файла отчетливо прослеживаются последовательности значения байт, начиная с «00» (на картинке отмечены красной рамкой). При этом, некоторые значения байтов выбиваются из данной последовательности. Можно сделать предположение, что выделенные байты в исходном файле были равны значению «00». Согласно правилу шифрования с использованием операции «побитового исключающего ИЛИ» («⊕»):

$$K \oplus 0 = K .$$

Следовательно, можно сделать вывод, что в качестве ключа использовался последовательный циклический счетчик байтов от «00» до «FF» включительно. Или другими словами, ключом шифрования каждого байта является его номер по модулю 256.

Для проверки данного предположения необходимо написать программу, осуществляющую расшифрование файла. Процедура расшифрования файла симметричная к процедуре шифрования, то есть:

$$T \oplus K = C$$

$$C \oplus K = T$$

$$\text{или } T \oplus K \oplus K = T \text{, где}$$

T – исходное значение байта,
K – ключ шифрования,
C – зашифрованное значение байта.

Пример такой программы на языке C++ представлен на листинге.

Листинг. Программа расшифрования файла

```
int main()
{
    // файл на чтение
    FILE* fin = fopen("pic_v1.bmp", "rb");
```

```
    // файл на запись результата
    FILE* fout = fopen("pic_v1_dec.bmp", "wb");
    // счетчик байтов - ключ
    unsigned char key = 0;

    unsigned char buf;

    // чтение первого байта в буфер BUF из файла FIN
    fread(&buf, 1, 1, fin);

    // цикл, пока не конец файла FIN
    while (!feof(fin))
    {
        // выполнение операции XOR между считанным байтом BUF
        // и ключом-счетчиком KEY
        buf = buf ^ key;
        // запись результата в файл FOUT
        fwrite(&buf, 1, 1, fout);
        // увеличение значения ключа-счетчика
        key++;
        // чтение следующего байта из файла FIN
        fread(&buf, 1, 1, fin);
    }  // while

    // Закрытие файлов
    fclose(fout);
    fclose(fin);
    return 0;
}
```

В результате выполнения программы получается новый файл “pic_v1_dec.bmp”, который открывается в редакторе изображений.


Рисунок 2 – Расшифрованный BMP-файл

Предположение на счет алгоритма генерации ключа шифрования оказалось верным.

**Ответ: 1) “MAXIMUM”**
**2) Ключ шифрования равен номеру байта по модулю 256 ИЛИ ключом шифрования является циклический счетчик от “00” до “FF” включительно.**

---

## Задача 3. Контроль доступа

Система охраны осуществляет удаленный учёт прохода сотрудников на предприятие с использованием карт сотрудников и контрольной суммы. Для внесения в электронный журнал записи о проходе сотрудника вычисляется контрольная сумма на основе текущей даты, фамилии сотрудника и серийного номера карты.

Для получения контрольной суммы, в первую очередь, вычисляется последовательность $(t_1 t_2 \dots t_p)$ по формуле:

$$t_1 = 1,$$
$$t_{i+1} = (t_i * d + m)\ mod\ N,$$

где $i = 0, 1, \ldots, p-1$ ,
$p$ – количество символов в фамилии сотрудника,
$d$ – текущая дата (день месяца),
$m$ – номер текущего месяца,
$N$ – количество символов в алфавите (для русского алфавита N=32) ,
*mod* – операция получения остатка от деления числа.

Далее каждый номер символа фамилии умножается на соответствующий элемент последовательности. Контрольная сумма вычисляется из суммы полученных произведений, умноженной на серийный номер карты сотрудника:

$$CRC = \left(t_1 * L_1 + t_2 * L_2 + \cdots + t_p * L_p\right) * Serial \bmod V,$$

где
$V$ – общее количество сотрудников ($V = 10\,000$);
*Serial* – серийный номер карты сотрудника (десятичное число в диапазоне от 1000 до 9999 включительно),
$L_i$ – номер $i$-го символа фамилии сотрудника в алфавите:

| **А** | **Б** | **В** | **Г** | **Д** | **Е** | **Ж** | **З** | **И** | **Й** | **К** | **Л** | **М** |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
| **Н** | **О** | **П** | **Р** | **С** | **Т** | **У** | **Ф** | **Х** | **Ц** | **Ч** | **Ш** | **Щ** |
| 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 |
| **Ъ** | **Ы** | **Ь** | **Э** | **Ю** | **Я** | | | | | | | |
| 26 | 27 | 28 | 29 | 30 | 31 | | | | | | | |

Например, если сотрудник по фамилии ПЕТРОВ пройдет через пропускной пункт 10 мая по карте с серийным номером 1350, то контрольная сумма будет вычислена следующим образом.

Последовательность $t$ будет состоять из 6 значений:

$t_1 = 1$ ,
$t_2 = (1 * 10 + 5) \bmod 32 = 15$ ,
$t_3 = (15 * 10 + 5) \bmod 32 = 27$,
$t_4 = (27 * 10 + 5) \bmod 32 = 19$ ,
$t_5 = (19 * 10 + 5) \bmod 32 = 3$,
$t_6 = (3 * 10 + 5) \bmod 32 = 3$ .

Контрольная сумма равна:

$$CRC = (1 * 15 + 15 * 5 + 27 * 18 + 19 * 16 + 3 * 14 + 3 * 2) * 1350 \bmod 10000 = 928 * 1350 \bmod 10000 = 2800.$$

Конкурент хочет проникнуть на предприятие под фамилией ВИЛКИН. Ему удалось добыть фрагмент журнала входа, а также имеется оборудование по программированию карт входа.

```
01.06 - ВИЛКИН - 9038
02.06 - ВИЛКИН - 2262
03.06 - ЛОЖКИН - 5066
04.06 - ЛОЖКИН - 5955
05.06 - ВИЛКИН - 5106
06.06 - ВИЛКИН - 1174
07.06 - ЛОЖКИН - 1462
08.06 - ЛОЖКИН - 4867
09.06 - ВИЛКИН - 6102
10.06 - ВИЛКИН - 5158
11.06 - ЛОЖКИН - 7858
12.06 - ЛОЖКИН - 3779
```

Какой серийный номер записать на карту, чтобы успешно пройти на предприятие? В какие дни лучше всего посетить предприятие, чтобы не вызвать подозрений. Ответ обоснуйте.

*К задаче прилагается:*
*файл журнала log_v1.txt.*

## Решение

Самый простой способ поиска серийного номера – перебор с вычислением контрольной суммы и сравнением с сохраненной в журнале посещений при известных других параметрах (дата, фамилия). Серийный номер карты лежит в диапазоне от 0 до 9999.

Суть алгоритма перебора: в цикле перебирается серийный номер. Для каждого значения вычисляется контрольная сумма по всем записям из журнала. Если контрольная сумма совпала со значением из журнала, значение серийного номера выводится на экран.

Пример такой программы на языке программирования C++ приведён в листинге.

Листинг. Перебор серийных номеров карты

```
// алфавит
char Letters[] = "АБВГДЕЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯ";
// размер алфавита
const int N = 32;
// общее количество сотрудников
const int V = 10000;

// Функция получения индекса буквы из алфавита
// Параметр:
//           LETTER - буква-символ
// RETURN индекс из массива Letters
int GetIndex(char letter)
{
   return strchr(Letters, letter) - Letters;
}

// Функция подсчета контрольной суммы
// Параметры:
//           FIO - фамилия сотрудника
//           DAY - дата прохода (день)
//           MONTH - дата прохода (месяц)
//           SERIAL - серийный номер карты сотрудника
// RETURN - вычисленное число - контрольная сумма
int GetCRC(char* fio, int day, int month, int serial)
{
   // длина фамилии
   int len = strlen(fio);
   int crc = 0;
   // создание массива T для вычисления последовательности t1..tp
   int* t = new int[len];
   // первый элемент последовательности
   t[0] = 1;

   // вычисление остальных элементов последовательности
   for (int i = 1; i < len; i++)
   {
        t[i] = (t[i - 1] * day + month) % N;
   }

   // вычисление контрольной суммы из последовательности T
   // и индексов букв фамилии
```

```
    for (int i = 0; i < len; i++)
    {
        crc += t[i] * GetIndex(fio[i]);
    }
    // умножение на серийный номер и взятие модуля
    crc = (crc * serial) % V;
    // возвращение вычисленного значения
    return crc;
}

int main()
{
    // структура, в которой содержится информация из журнала
    struct record {
        int day;        // день
        int month;      // месяц
        char fio[10];   // фамилия
        int crc;        // контрольная сумма
    };

    const int logSize = 6;
    // массив записей из журнала
    // взяты только записи ВИЛКИНА
    record log[logSize] = {
    {1, 6, "ВИЛКИН", 9038},
    {2, 6, "ВИЛКИН", 2262},
    {5, 6, "ВИЛКИН", 5106},
    {6, 6, "ВИЛКИН", 1174},
    {9, 6, "ВИЛКИН", 6102},
    {10, 6, "ВИЛКИН", 5158}
    };

    int serial;
    int successCount;

    // цикл перебора серийного номера для карты ВИЛКИНА
    for (serial = 0; serial < 10000; serial++)
    {
        // счетчик совпадений с записями из журнала
        successCount = 0;
        for (int i = 0; i < logSize; i++)
        {
            // если вычисленный CRC совпал с хранимым в журнале
            // увеличивается счетчик совпадений
            if (GetCRC(log[i].fio, log[i].day, log[i].month, serial)
== log[i].crc)
                successCount++;
        }
        // если все записи журнала совпали
        // (счетчик совпадений равен размеру журнала)
        // вывод на экран серийного номера
        if (successCount == logSize)
            cout << serial << endl;
    }
    return 0;
}
```

В результате выполнения программы первое вычисленное значение равно 1327. Это и есть серийный номер карты ВИЛКИНА.

Если проанализировать журнал прохода, можно заметить, что ВИЛКИН и ЛОЖКИН ходят на работу в режиме 2 дня через 2 дня. Поэтому для прохода на предприятие лучше выбрать те дни, в которые ВИЛКИН должен посетить работу: то есть, 15-16 июня, 19-20 июня и так далее.

**Ответ:**

**1) серийный номер карты ВИЛКИНА – 1327.**

**2) ВИЛКИН ходит на работу 2 дня через 2. Посещать предприятие надо в эти же дни, чтобы не вызывать подозрение.**

---

## Задача 4. Reverse engineering

Имеется фрагмент программы на языке C:

```
#include <stdio.h>
#include <string.h>

int main ()
  {
    int code = 0;
    char password[10];
    printf ("Введите пароль:");
    gets (password);
    /*
    ...
    утерянный фрагмент кода
    ...
    */
    if (code == 0)
    {
      /*
      ...
      вычисление значения code
      ...
    */
    }

    if (code == 21827)
       printf ("Пароль верный!");
    else
       printf("Пароль неверный!");
  }
```

Был получен фрагмент скомпилированного исполняемого файла в шестнадцатеричном формате, в котором удалось обнаружить следующий программный код:

| Адрес памяти | Байты в памяти | Их содержание |
|---|---|---|
| 001C1231 | C2 E2 E5 E4 E8 F2 E5 20 EF E0 D0 EE EB FC 3A 00 | Строка «Введите пароль:» |
| 001C1241 | CF E0 D0 EE EB FC 20 E2 E5 F0 ED FB E9 21 00 | Строка «Пароль верный!» |
| 001C1250 | CF E0 D0 EE EB FC 20 | Строка «Пароль неверный!» |

| | | |
|---|---|---|
| | ED E5 E2 E5 F0 ED FB E9 21 00 | |
| 001C1261 | 00 00 00 00 00 00 00 00 00 00 | Массив password (10 байт) |
| 001C126B | 00 00 00 00 | Переменная code (целое число, 4 байта) |
| 001C126F | 68 61 12 1C 00 | Машинная команда, передающая параметр password (его адрес) в подпрограмму gets (переход по адресу начала тела подпрограммы) |
| 001C1274 | FF 75 15 1C 00 | Вызов подпрограммы gets |
| 001C1279 | 83 C4 04 | Вспомогательная команда, выполняемая после возвращения из подпрограммы, имеющей один параметр |
| 001C127C | 8B 45 6B 12 1C 00 | Копирование значения переменной code в регистр процессора |
| 001C1282 | 39 45 00 | Сравнение значения регистра с константой 0 |
| 001C1285 | 76 11 13 1C 00 | Если результат сравнения false, то переход на выполнение команды по адресу 00 1C 13 11 |
| 001C128A | ... | Вычисление переменной code |
| ... | ... | |
| 001C1311 | 8B 45 6B 12 1C 00 | Копирование значения переменной code в регистр процессора |
| 001C1317 | 3B 45 43 55 | Сравнение значения регистра с константой 21827 |
| 001C131A | 77 2C 13 1C 00 | Если результат сравнения true, то переход по адресу 00 1C 13 2C |
| 001C131F | 68 50 12 1C 00 | Машинная команда, передающая адрес строки «Пароль неверный!» в подпрограмму printf как параметр |
| 001C1324 | FF 90 16 1C 00 | Вызов подпрограммы printf |
| 001C1329 | 83 C4 04 | Вспомогательная команда, выполняемая после возвращения из подпрограммы, имеющей один параметр |
| 001C132B | C3 | Команда завершения работы программы |
| 001C132C | 68 41 12 1C 00 | Машинная команда, передающая адрес строки «Пароль верный!» в подпрограмму printf как параметр |
| 001C1331 | FF 90 16 1C 00 | Вызов подпрограммы printf |
| 001C1336 | 83 C4 04 | Вспомогательная команда, выполняемая после возвращения из подпрограммы, имеющей один параметр |
| 001C1339 | C3 | Команда завершения работы программы |
| ... | ... | |
| 001C1575 | | Тело функции gets |
| ... | ... | |
| 001C168F | C1 | Возврат к команде, следующей после точки вызова |
| 001C1690 | | Тело функции printf |
| ... | ... | |
| 001C185F | C1 | Возврат к команде, следующей после точки вызова |

Определите, что нужно подать на вход программы, чтобы в результате выполнения вывелась строка «*Пароль верный!*». Ответ обоснуйте.

Применяемая реализация подпрограммы *gets* принимает на вход любые данные без ограничений. Специальные символы (нулевой байт, символ конца строки и т.п.) во входном потоке не прерывают ввод строки.

## Решение

Способ 1 – переполнение буфера

Из анализа фрагмента памяти можно определить, что в памяти массив password (10 байт) и переменная code (4 байта) следуют друг за другом. В условии сказано, что функция gets не осуществляет проверку на длину вводимых строковых данных.

Проанализировав исходный код программы и команды по работе с переменной code, можно сделать вывод, что переменная code НЕ БУДЕТ вычисляться, если она не равна нулю.

При этом, фраза «Пароль верный!» выводится на экран только в случае, когда значение переменной code равно $21827_{10}$ = 00 00 55 $43_{16}$ .

Следовательно, для вывода на экран фразы «Пароль верный!» необходимо, чтобы:

1) после ввода пароля переменная code не была нулевой,
2) в переменной code должно содержаться число $21827_{10}$ .

Если проанализировать фрагмент памяти и посмотреть, как хранятся переменные в памяти, можно увидеть, что используется кодировка Little Endian для хранения чисел. То есть порядок байт в числе должен быть обратным. Следовательно, для сохранения числа 21827 в переменной code, необходимо, чтобы в памяти оно было записано в обратном порядке следования байт: `43 55 00 00`.

Для записи значения `43 55 00 00` в переменную code необходимо при вводе пароля ввести любые 10 символов (заполнение массива password), а затем передать числа $43_{16}$ и $55_{16}$, Воспользовавшись ASCII-таблицей определяем символы по указанным значениям:

$43_{16}$ – ‘C’,

$55_{16}$ – ‘U’.

Ответ: *любые 10 символов + ‘CU’. Нулевые байты в конце передавать необязательно.*

Способ 2. Выполнение функции printf

Вывод фразы «Пароль верный!» на экран осуществляется следующим образом:

1) Передача адреса строки «Пароль верный!» в подпрограмму printf (`68 41 12 1C 00`).
2) Вызов подпрограммы printf (`FF 90 16 1C 00`).
3) Выполнение вспомогательной команды после возвращения из подпрограммы, имеющей один параметр (`83 C4 04`).
4) Завершение работы программы (`C3`).

Следовательно, после ввода пароля необходимо добиться такого содержимого в памяти, чтобы запустилась программа printf с параметром «Пароль верный!».

Данная последовательность команд должна запускаться после выполнения подпрограммы gets, по адресу `001C127C` (вместо копирования значения переменной code в регистр процессора).

Поэтому при вводе пароля необходимо передать следующую строку:

1) Любые 10 символов (массив password).
2) Любые 4 символа (переменная code).
3) Любые 5 байт вместо команды передачи параметра подпрограмме gets (`68 61 12 1C 00`).
4) Любые 5 байт вместо команды вызова подпрограммы gets (`FF 75 15 1C 00`).
5) Байты возвращения из подпрограммы с одним параметром (`83 C4 04`).
6) Передача адреса строки «Пароль верный!» в подпрограмму printf (`68 41 12 1C 00`).

7) Вызов подпрограммы printf (`FF 90 16 1C 00`).
8) Выполнение вспомогательной команды после возвращения из подпрограммы, имеющей один параметр (`83 C4 04`).
9) Завершение работы программы (`C3`).

Стоит отметить, что до пункта 4 включительно программа уже выполнилась, и содержимое памяти по этим адресам не представляет интереса. Поэтому можно записывать любые значения.

Ответ: *любые 24 символа* + *`83 C4 04 68 41 12 1C 00 FF 90 16 1C 00 83 C4 04 C3`*.

**Ответ: любые 10 символов + ‘CU’**
**ИЛИ**
**любые 24 символа + 83 C4 04 68 41 12 1C 00 FF 90 16 1C 00 83 C4 04 C3**

---

## Задача 5. Web-сайт

Олег создал сайт, в котором спрятал IP-адрес своего секретного сервера в формате `xxx.xxx.xxx.xxx` (`xxx` – число от 0 до 255). На сайте Олег оставил подсказки. Определите IP-адрес секретного сервера Олега.

*К задаче прилагается:*
*папка с содержимым web-сайта*.

## Решение

IP-адрес выглядит: xxx.xxx.xxx.xxx (4 числа, разделенных точками).
Ищем подсказки:

1) В файле index.html – `<!--End META 192. block -->` - начало IP-адреса: «192.»
2) Файл style.css – странный комментарий `/* 0x183549 */`. Если посмотреть картинку (img.png) в Hex-редакторе, то по указанному адресу будет фраза – «.64.» – вторая компонента (так как встречается раньше).
3) Там же в файле style.css второй странный комментарий – `/* 20 C6 D4 04 */`. Поищем комбинацию этих байтов в картинке img.png. Она встречается всего 1 раз, и после нее содержится фраза – «.118.» – третья компонента
4) Файл script.js – есть странный комментарий – `/* Divide .50 */` – «.50» – конец IP-адреса

Учитывая формат найденных чисел и последовательность расставленных подсказок, можно сделать следующий вывод:
“192.” – начало IP-адреса,
“.64.” – второй байт IP-адреса (встречается раньше других подсказок),
“.118.” – третий байт IP-адреса (подсказка, указывающая на этот байт встречается после предыдущей подсказки),
“.50” – конец IP-адреса.

**Ответ: 192.64.118.50**

---