



Название:	Таск древних русов
Категория:	Реверс-инжиниринг
Уровень:	Средняя
Очки:	550
Описание:	В архиве всё это время скрывалась правда о великой битве. А самое главное - сохранились файлы того времени! Только вот часть информации была утеряна...
Теги:	Побайтовое шифрование, патчинг файла
Автор:	ROP

Прохождение:

Изучаем данные нам файлы.

The screenshot shows a Windows desktop environment. In the center, there are two instances of the Code::Blocks IDE. The top window displays the file `not_important.c`, which contains C code related to ancient Slavic grammar. The bottom window displays the file `ancientslavic.h`, which contains C preprocessor definitions for various Slavic words and grammatical forms. To the left of the IDE windows, a Notepad window titled "change.log" is open, showing a log of changes made to the files. The desktop background is blue, and several icons are visible on the taskbar and desktop.

```

not_important.c - Code::Blocks 20.03
File Edit View Search Project Build Debug Fortran wxSmith Tools Plugins Doxygen Settings Help
Management Start here not_important.c
Projects Files FSymbols Workspace
1 #include <stdio.h>
2 #include <string.h>
3 #include "ancientslavic.h"
4
5    добръ струже прав[] стане чередъ броны1, броны2, броны3, броны4, броны5, броны6, брон
6    молитва отченам
7
8    добръ струже грамота[корок] стане чередъ дуло альянъ ъ
9    добръ струже старый стане дуло ъ
10   добръ цело четь, многи ъ
11
12
13   молви нека "Дорогий князь смила замина: " болванъ ъ
14   вислови грамота, корок, говор буде
15   многи стане дали грамота буде

ancientslavic.h
1 // Снова мелочи жизни
2 #define старый last_c
3 #define грамота грамота
4 #define корок 40
5 #define семъ 7
6 #define четь 3
7 #define левяль 9
8 #define чередъ {
9 #define говор stdin
10 #define вет i
11 #define ёще ++
12 #define струже char
13 #define зодръ unsigned
14 #define прав correct
15 #define болванъ )
16 #define нека [
17 #define многи len
18 #define дали strlen(
19
20 #define броны1 0x71
21 #define броны2 0x33
22 #define броны3 0x4d
23 #define броны4 0x65

C:\Users\Iffa\Desktop\ancientslavic.h - Notepad+ [Administrator]
File Правка Поиск Вид Кодировки Синтаксисы Опции Инструменты Макросы Запуск Плагины Вкладки ?
length:1267 lines:53 Ln:1 Col:1 Pos:1 Unix (LF) UTF-8 INS

```

Нам интересен Си-файл и Н-файл.

Если попробовать поискать этот язык по всему коду, то попадём сюда:

#include <stdio.h>#include <string.h>#include "ancientslavic.c"

поиск алиса картинки видео карты товары финансы квартиры переводчик все

Быть может, вы искали «#include <stdio.h>#include <string.h>#include "ancientslavic.h" «**добра** стружие прав[] стане чередъ бронь1, бронь2, бронь3, бронь4, бронь5, бронь6, бронь7, бронь8, бронь9, бронь10, бронь11, бронь12, бронь13, бронь14, бронь15, бронь16, бронь17, бронь18, бронь19, бронь20, бронь21, бронь22, бронь23, бронь24, бронь25, бронь26, бронь27, бронь28, бронь29, бронь30, бронь31, бронь32 аминь ъмолитва отченашдобр»?

Старославянский Сиъ | AlterVision | Антон AlterVision Резниченко
av13.ru › notes/ancient-slavic-c/
Старославянский Сиъ. Опубликовано 12.12.2010 в 12:46 в разделе Важное, Заметки. ...
Чтобы сделать из своего любимого языка такую дребедень, нам понадобится один особый файл — **ancientslavic.h** — код которого представлен ниже.

<https://www.av13.ru/notes/ancient-slavic-c/>

Тут есть дополнительные слова для H-файла:

```
// Файл: ancientslavic.h - переопределение языка

// Мелочи жизни
#define поболе >=
#define помене <=
#define боле >
#define мене <
#define аки ==
#define стане =
#define да +
#define без -
#define раз *
#define дели /
#define ъ ;
#define право true
#define бреше false
#define дулю 0

// if () { } else if () { } else { }
#define коли if(
#define пущай ){
#define ежели }else if(
#define либо }else{
```

```
#define аминь }
// коли а аки 1 пущай в стане 1 ежели а аки 2 пущай в стане 4 либо стане 0 аминь

// while () {} do{} while() for {}
#define покуда while(
#define твори do{
#define доколе }while(
#define буде );
#define откель for(
#define ступай goto
#define вон exit
#define

// покуда а мене 10 пущай а стане а да 1 аминь
// твори а стане а без 1 доколе а поболе 0 буде

// int float char etc
#define цело int
#define дробно float
#define передробно double
#define азъ char
#define непотребо void

// I/O
#define молви printf
#define поведай scanf

// Процедура - это функция, которая нам ничего не даёт, отсюда:
#define молитва int
#define мольба void
#define отченаш main(){
#define воздати return
// молитва отченаш воздати 0 ъ аминь
```

Можно заменить каждое слово из таска на значение из H-файла.

добръ стружие старый стане дулю ъ
добръ цело чет, многи ъ

молви межа "Поведай мне одинъ тайна: " болванъ ъ
вымолви грамота, сорок, говор буде
многи стане даль грамота буде

откель чет стане дулю ъ чет мене многи ъ чет вяче пущай
грамота [чет] стане грамота [чет] ворожба три;
грамота [чет] стане грамота [чет] да семь;
грамота [чет] стане грамота [чет] да семь;
грамота [чет] стане грамота [чет] без девять;
грамота [чет] стане грамота [чет] ворожба девять;
грамота [чет] стане грамота [чет] да чет;
грамота [чет] стане грамота [чет] ворожба чет;
грамота [чет] стане грамота [чет] ворожба чет;
грамота [чет] стане грамота [чет] ворожба старый;

старый стане грамота [чет] ъ
аминь

коли быти грамота, прав, многи болванъ аки дулю пущай

Например, вот тут будет шифрование введённого флага. Методом исключения выясним, что для слово **ворожба** подходит xor (так как его результат можно обратить, а +/- уже заняты).

Напишем код для дешифрования:

```
#include <stdio.h>
unsigned char correct[] = {
    0x71, 0x33, 0x4d, 0x65,
    0x11, 0x27, 0xa2, 0xee,
    0x9e, 0xf2, 0xc9, 0xa3,
    0xd7, 0x55, 0x32, 0x43,
    0x3b, 0x56, 0x6, 0x87,
    0xe0, 0x9b, 0xf9, 0x86,
    0xdf, 0x47, 0xc1, 0xb5,
    0xef, 0x7f, 0xff, 0xd9
};

int main() {
    int len = sizeof(correct);
    unsigned char last = 0;
    int i;

    for(i = 0; i < len; i++) {
```

```

unsigned char temp, j;

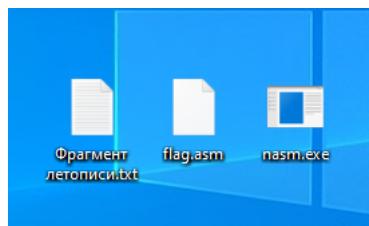
for (j = 0x20; j < 128; j++) {
    temp = j;

    /* Алгоритм шифрования */
    temp = temp ^ 3;
    temp = temp + 7;
    temp = temp + 7;
    temp = temp - 9;
    temp = temp ^ 9;
    temp = temp + i;
    temp = temp ^ i;
    temp = temp ^ i;
    temp = temp ^ last;

    if (temp == correct[i]) {
        last = temp;
        printf("%c", j);
        break;
    }
}
printf("\n");
return 0;
}

```

Получили пароль от архива - `p@s$w0rD_f0R_tHe_S1aViC_@rcH1ve`.



Здесь `flag.asm` - это ложный вектор.

А вот `nasm.exe` пропатчен. Понять это можно, если скачать `nasm.exe` той же версии и сравнить его.

```
C:\Users\fff\Desktop>nasm.exe -v
NASM version 2.16.01 compiled on Dec 21 2022
```

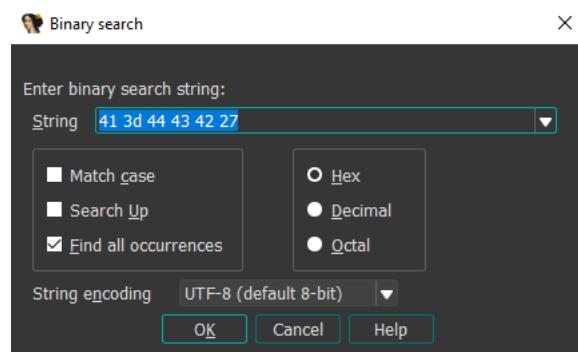
На сайте это:

<https://nasm.us/pub/nasm/releasebuilds/2.16.01/win64/nasm-2.16.01-win64.zip>

```
C:\Users\fff\Desktop>fc /b nasm.exe nasm_orig.exe
Сравнение файлов nasm.exe и NASM_ORIG.EXE
0004ECB0: 41 00
0004ECB1: 3D 00
0004ECB2: 44 00
0004ECB3: 43 00
0004ECB4: 42 00
0004ECB5: 27 00
0004ECB6: 49 00
0004ECB7: 3F 00
0004ECB8: 5E 00
0004ECB9: B0 00
0004ECBA: 9F 00
0004ECBB: 0F 00
0004ECBC: 63 00
0004ECBD: 3E 00
0004ECBE: 54 00
0004ECBF: 2D 00
0004ECC0: 54 00
0004ECC1: 40 00
0004ECC2: 51 00
0004ECC3: 39 00
0004ECC4: 2D 00
0004ECC5: 66 00
```

Найдём эти байты в файле через IDA.

Жмём ALT+B и ищем по:



Или по другой последовательности байтов из данных выше.

В итоге попадём сюда.

```

.data:00000001400502A4          db    0
.data:00000001400502A5          db    0
.data:00000001400502A6          db    0
.data:00000001400502A7          db    0
.data:00000001400502A8 qword_1400502A8 dq 0 ; DATA X
| db 41h ; A
.data:00000001400502B0          | db 3Dh ; =
.data:00000001400502B1          db 44h ; D
.data:00000001400502B2          db 43h ; C
.data:00000001400502B3          db 42h ; B
.data:00000001400502B4          db 27h ; '
.data:00000001400502B5          db 49h ; I
.data:00000001400502B6          db 3Fh ; ?
.data:00000001400502B7          db 5Eh ; ^
.data:00000001400502B8          db 0B0h
.data:00000001400502B9          db 9Fh
.data:00000001400502BA          db 0Fh
.data:00000001400502BB          db 63h ; c
.data:00000001400502BC          db 3Eh ; >
.data:00000001400502BD          db 54h ; T
.data:00000001400502BE          db 2Dh ; -
.data:00000001400502BF          db 54h ; T
.data:00000001400502C0          db 40h ; @
.data:00000001400502C1          db 51h ; Q
.data:00000001400502C2          db 39h ; 9
.data:00000001400502C3          db 2Dh ; -
.data:00000001400502C4          db 66h ; f
.data:00000001400502C5          db 52h ; R
.data:00000001400502C6          db 10h
.data:00000001400502C7          db 3Bh ; ;
.data:00000001400502C8          db 2Dh ; :
0004ECB0 00000001400502B0: .data:00000001400502B0 (Synchronized with Hex View-1)

```

Можем попробовать представить эти байты как код, но ничего не выйдет. Нужно спустится ниже после некоторого пробела из нулей.

```
.data:00000001400502CC          db 12h
.data:00000001400502CD          db 5Fh ; _
.data:00000001400502CE          db 54h ; T
.data:00000001400502CF          db 10h
.data:00000001400502D0          db 45h ; E
.data:00000001400502D1          db 52h ; R
.data:00000001400502D2          db 5Fh ; _
.data:00000001400502D3          db 64h ; d
.data:00000001400502D4          db 4Bh ; K
.data:00000001400502D5          db 0
.data:00000001400502D6          db 0
.data:00000001400502D7          db 0
.data:00000001400502D8          db 0
.data:00000001400502D9          db 0
.data:00000001400502DA          db 0
.data:00000001400502DB          db 0
.data:00000001400502DC          db 0
.data:00000001400502DD          db 0
.data:00000001400502DE          db 0
.data:00000001400502DF          db 0
.data:00000001400502E0          db 0B9h |
.data:00000001400502E1          db 25h ; %
.data:00000001400502E2          db 0
.data:00000001400502E3          db 0
.data:00000001400502E4          db 0
.data:00000001400502E5          db 48h ; H
.data:00000001400502E6          db 0BEh
.data:00000001400502E7          db 0B0h
.data:00000001400502E8          db 2
.data:00000001400502E9          db 5
.dta:00000001400502EA          db 10h ; @
0004ECEO 00000001400502E0: .data:00000001400502E0 (Synchronized with Hex View-1)
```

Ставим курсор как на скрине и жмём "С".

```

.data:00000001400502D5          db    0
.data:00000001400502D6          db    0
.data:00000001400502D7          db    0
.data:00000001400502D8          db    0
.data:00000001400502D9          db    0
.data:00000001400502DA          db    0
.data:00000001400502DB          db    0
.data:00000001400502DC          db    0
.data:00000001400502DD          db    0
.data:00000001400502DE          db    0
.data:00000001400502DF          db    0
.data:00000001400502E0 ; -----
.data:00000001400502E0          mov   ecx, 25h ; '%'
.data:00000001400502E5          mov   rsi, 1400502B0h
.data:00000001400502EF
.data:00000001400502EF loc_1400502EF:           ; CODE XREF: .data:000
.data:00000001400502EF          cmp   ecx, 0
.data:00000001400502F2          jz    short locret_140050308
.data:00000001400502F4          dec   ecx
.data:00000001400502F6          mov   al, [rsi]
.data:00000001400502F8          add   al, 99h
.data:00000001400502FA          xor   al, 99h
.data:00000001400502FC          mov   [rsi], al
.data:00000001400502FE          inc   esi
.data:0000000140050300          jmp   short loc_1400502EF
.data:0000000140050300 ; -----
.data:0000000140050302          db    0
.data:0000000140050303          db    0
.data:0000000140050304          db    0
.data:0000000140050305          db    0

```

Этот код дешифрует байты выше.

Можем сделать это сами:

[https://cyberchef.org/#recipe=From_Hex\('Auto'\)ADD\({'option':'Hex','string':'99'}\).XOR\({'option':'Hex'}\)](https://cyberchef.org/#recipe=From_Hex('Auto')ADD({'option':'Hex','string':'99'}).XOR({'option':'Hex'}))

В флаге остаётся непонятная часть.

CODEBY{AnĐi1eNt_

На самом деле это из-за кодировки. Попробовав разные кодировки поймём, что тут русский символ С.

Тогда начало флага будет:

CODEBY{AnC1eNt...