



Название:	Три Лика Тени
Категория:	Реверс-инжиниринг
Уровень:	Средний
Очки:	1000
Описание:	<p>Расклад "Три Лика Тени"</p> <p>(Карты легли на стол странным узором...)</p> <p>1. Правящая Сила (Перевёрнутая) "То, что кажется защищой — лишь маска. Ищи там, где стражник поворачивается спиной. Его меч ржавеет от лунного света..."</p> <p>2. Тройка Мечей "Три удара в пустоту. Третья рана — невидима. Она в том, что никогда не истекает кровью, но всегда открыта..."</p> <p>3. Отшельник (В зеркальном отражении) "Одиночный свет во тьме. Не следуй за ним — он ведёт к обрыву. Истина в его тени, что тянется назад, к началу всех дорог..."</p> <p>4. Пятерка Пентаклей "Монеты с выщербленными краями. Они звенят фальшиво. Настоящее богатство замуровано в стене — там, где камень теплеет под пальцами..."</p> <p>5. Башня (Покрытая инеем) "Высокие врата рухнули. Но в подземелье, куда не ступала нога, дрожит последний факел. Его свет пишет слово на влажных камнях..."</p> <p>Совет Духов: "Когда увидишь девять воронов — считай до десяти. Когда найдёшь все двери — стучи в стену. Когда прочтёшь все надписи — смотри на промежутки между буквами."</p> <p>Теги:</p> <p>Патчинг, обfuscация</p>

Прохождение :

Весь код в `main` и арт в ТХТ-файле лишь отвлекающий элемент. Описание таска и абсурдность всего в функции `main` явно намекают на это. Посмотрим, что выполняется после `main` в отладчике.

The screenshot shows the Immunity Debugger interface with three assembly windows:

- Top Window:** Shows assembly from address `.text:00000000004013B4` to `.text:00000000004013D5`. It includes instructions like `mov r8, cs:envp ; envp`, `mov rdx, cs:argv ; argv`, `call main`, and `jnz loc_40149E`.
- Middle Left Window:** Shows assembly from address `.text:00000000004013DB` to `.text:00000000004013E3`. It includes `mov edx, cs:has_ctor`, `test edx, edx`, and `jnz short loc_4013F0`.
- Middle Right Window:** Shows assembly from address `.text:000000000040149E` to `.text:00000000004014A0`. It includes `loc_40149E: ; Code`, `mov ecx, eax`, and `call exit`.

A red arrow points from the `loc_4013F0` label in the middle left window to the `loc_40149E` label in the middle right window, indicating a jump from the left section to the right section.

В этот момент обычно программы идут направо. А тут код пропатчен и ведёт к левому блоку.

```
.text:00000000004013E3 jnz      short loc_4013F0
.text:00000000004013E5 call     _cexit
.text:00000000004013EA mov      eax, cs:mainret
.text:00000000004013F0
.text:00000000004013F0 loc_4013F0:
.text:00000000004013F0 add      rsp, 98h
.text:00000000004013F7 pop     rbx
.text:00000000004013F8 pop     rsi
.text:00000000004013F9 pop     rdi
.text:00000000004013FA pop     rbp
0007E5 0000000000004013E5: __tmainCRTStartup+265 (Synchronized with RIP)
```

Эта функция пропатчена.

```
.text:00000000004054C8
.text:00000000004054C8
.text:00000000004054C8 ; Attributes: thunk
.text:00000000004054C8
.text:00000000004054C8 ; void __cdecl _cexit()
.text:00000000004054C8 public _cexit
.text:00000000004054C8 _cexit proc near
.text:00000000004054C8 jmp     _cexit_0
.text:00000000004054C8 _cexit endp
.text:00000000004054C8
```

```
.idata:000000000040B9D0 ; void __cdecl cexit_()
idata:000000000040B9D0 _cexit_0 proc near
idata:000000000040B9D0 mov    rax, 40B9B0h
idata:000000000040B9D7 mov    rcx, 0

idata:000000000040B9DE loc_40B9DE:
idata:000000000040B9DE sub    byte ptr [rax+rcx], 69h ; 'i'
idata:000000000040B9E2 inc    rcx
idata:000000000040B9E5 cmp    byte ptr [rax+rcx], 0
idata:000000000040B9E9 jnz    short loc_40B9DE

idata:000000000040B9EB mov    rax, 0
idata:000000000040B9F2 mov    eax, 40000h
idata:000000000040B9F7 add    eax, 54A0h
idata:000000000040B9FC push   rax
idata:000000000040B9FD retn
idata:000000000040B9FD _cexit_0 endp ; sp-analysis failed
```

Код дешифровки флага.

```
; DATA XREF: .idata:000000000040B034 to
idata:000000000040B8A7 align 8
idata:000000000040B8A8 dq 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
idata:000000000040B960 dq 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
idata:000000000040B9B0 dq 306C5F417B435343h, 7440645F666F5F74h, 5F315F644E615F61h, 7D67406C66h
idata:000000000040B9D0
idata:000000000040B9D0 ; ===== SUBROUTINE =====
idata:000000000040B9D0
```

Так он будет выглядеть после дешифровки. Останется лишь представить в виде строки.