****
Приложение к статье.

Материалы по работе виртуальной машины SecuRom версии 7.33.0017
Все данные были получены при дисассемблировании файла cnc3game.dat (NODVD), который содержал VM пристроенную в виде секции .memory
Файл был взят с ресурса: http://www.playground.ru/cheats/4932/
В данной редакции файла реверс-инженерами в виртуальную машину были внесены изменения:
Для корректного декодирования смещений в “Хранилище № 1.5” на всех машинах во всех островках две инструкции
CPUID
AND EAX,FFFFFFDF
заменены на
MOV EAX,40F92
т.е. ключ для инструкции ROR, расшифровывающей смещения один и то же = 92h

Неизменяемая структура главного хранилища:
(+0) Хранилище_1.5 = 01FD1000
(+C) ВХОД В ВИРТУАЛЬНУЮ МАШИНУ = 020A5000
(+14) ВХОД В ГЛАВНОЕ ХРАНИЛИЩЕ = 020A7000
(+20) Хранилище В СЕКЦИИ .ARTEM = 01212A10
(+28) Хранилище №2 = 020A6000
}
Действующий переходник в VM для баз запросов:
00D44F40 JMP DWORD PTR DS:[12E043C]

Все упомянутые в документе островки без обфускации!
В документе:
Способ №1. Вся цепь вызова для
0040A00D CALL 0040A370
Способ №1A. Вся цепь вызова для
0044F669 CALL EAX SetUnhandledExceptionFilter
Способ №2. Часть островков для прямого хода алгоритма.
Информация по логике работы способа №2.
****

Иерархия в использовании регистров процессора на островках
Абсолютно закрепленные регистры:
EBX – Главное хранилище
CL(ECX) – ROL-байт
Плавающие регистры:
EDI – Ресурс (+10) в главном хранилище
ESI – Вспомогательный
EDI – Ячейки в главном хранилище
EAX/EDX/EDI/ESI – DWORD по виртуальному указателю (V-ESP) или DWORD в ячейках
EAX/EDX/EDI – В качестве операндов для прыжка на следующий островок

```
--------------------------------------------------------------------------
Способ №1.
0040A00D  CALL 0040A370
Стрелка:
0040A370   JMP DWORD PTR DS:[15000C4]
База запроса:
0157C830    PUSH OFFSET 0157C84A
0157C835    PUSH 0040365A
0157C83A    PUSH OFFSET 00D611CC
0157C83F    PUSHFD
0157C840    SUB DWORD PTR SS:[ESP+4],1C28C
0157C848    POPFD
0157C849   RETN
Обвертка виртуального стека для вызова:
0157C84A
Верхушка виртуального стека:
014EA729
Окончание виртуального стека:
01557450
Количество используемых DWORDs в виртуальном стеке:
12
Адрес запрашиваемой функции:
015110F0
--------------------------------------------------------------------------

      02B3B7CE
MOV EAX, DWORD PTR DS:[ (+4) ] ; EAX = 020A7004
ADD EAX, DWORD PTR DS:[ (+C) ] ; EAX = FF4B2420 + 020A5000 = 01557420
MOV EAX,DWORD PTR DS:[EAX]   ; EAX = 0522366C
MOV EDX,EAX ; EAX = 0522366C
MOV CL,BYTE PTR DS:[ (+10) ] ; EDI = 020A7010
PUSH ECX ; ECX = 0080206B
SHL EDX, 18   ;  EDX = 6C000000
SHR EDX, 18  ;  EDX =  0000006C
POP ECX  ; ECX = 0080206B
ADD BYTE PTR DS:[ (+10) ],DL ; EDI = 020A7010 ; DL = 6C
MOV EDX,DWORD PTR SS:[ESP] ; EDX = 0522366C
SHL EAX, 8 ; EAX = 22366C00
SHR EAX, 18 ; EAX = 00000022
POP ECX ; ECX = 0080206B
ROR AL,CL ; AL = 22 ; CL = 6B => EAX = 00000044
SHL EAX,2 ; EAX = 00000044 * 4 = 00000110
ADD EAX,EBX ; EAX = 00000110 + 020A7000 = 020A7110
MOV EAX,DWORD PTR DS:[EAX] ; EAX = 00000000
PUSH ECX ; ECX = 0080206B
SHR EDX, 18  ; EDX = 0522366C = > EDX = 00000005
POP ECX  ; ECX = 0080206B
ROL DL,CL ; DL = 05 ; CL = 6B  =>  DL = 28
SHL EDX,2  ; EDX = 00000028 * 4 = 000000A0
ADD EDX,EBX ; EDX = 020A7000 + 000000A0 = 020A70A0
MOV DWORD PTR DS:[EDX],EAX  ; EAX = 00000000
SHL EDX, 10 ; EDX = 0522366C => EDX = 366C0000
SHR EDX, 18 ; EDX = 00000036
POP ECX ; ECX = 0080206B
ADD DL,CL ; DL = 6B + 36 = A1
MOV EAX,DWORD PTR DS:[ (+0) ] ; EBX = 020A7000 => EAX = FFF2C000
ADD EAX,DWORD PTR DS:[ (+C) ] ; EAX = FFF2C000 + 020A5000 = 01FD1000
ADD DWORD PTR DS:[ (+4) ], 4  ; ECX = 020A7004
```

```
SHL EDX,2 ; EDX = 000000A1 * 4 = 00000284
ADD EDX,EAX ; EDX = 00000284 + 01FD1000 = 01FD1284
MOV EAX,DWORD PTR DS:[ (+C) ]  ; ECX = 020A700C ; => EAX = 020A5000
MOV EDX,DWORD PTR DS:[EDX] ; EDX = 01FD1284 ; => EDX = C003FFF1
MOV DWORD PTR SS:[ESP],EDX ; [ESP] = C003FFF1
PUSH EAX ;  EAX = 020A5000
CPUID
AND EAX,FFFFFFDF
MOV CL,AL ; CL = AL = 92
ROR DWORD PTR SS:[ESP+4],CL ; [ESP+4] = C003FFF1 / F0 = FFFC7000
POP EAX ; EAX = 020A5000
ADD DWORD PTR SS:[ESP],EAX ; [ESP] = FFFC7000 + 020A5000 = 0206C000
POP EAX ; EAX = 0206C000
JMP EAX

      0206C000
MOV DWORD PTR DS:[ (+4) ] ; EAX = 020A7004 ; => EAX = FF4B2424
ADD EAX,DWORD PTR DS:[ (+C) ] ; EAX = FF4B2424 + 020A5000 = 01557424
MOV DWORD PTR SS:[ESP],EAX  ;   EAX = 01557424
MOV EDX, EAX
ADD EDX,4 ;   EDX = 01557424 + 4 = 01557428
MOV EDX, DWORD PTR DS:[EDX] ; EDX = 01557428 ; EDX = 2159D5AC
MOV EAX, DWORD PTR DS:[EAX] ; EAX = 01557424 ; EAX = 889AFC25
PUSH EAX     ; EAX = 889AFC25
MOV CL,BYTE PTR DS:[ (+10) ] ; CL = 97
SHL EAX,18  ; EAX = 25000000
SHR EAX,18 ; EAX = 00000025
ADD BYTE PTR DS:[ (+10) ],AL ; (+10) = D7 + 25 = FC
POP EAX ; EAX = 889AFC25
PUSH EAX
ROR DL,CL ; DL = AC ; CL = D7 ; => DL = 59
ROR EDX,8 ; EDX = 2159D559 ; => EDX = 592159D5
ROR DL,CL ; DL = D5 ; CL = D7; => DL = AB
ROR EDX,8 ; EDX = AB592159
ROR DL,CL ; DL = 59 ; CL = D7 ; => DL = B2
ROR EDX,8 ; EDX = B2AB5921
ROR DL,CL ; DL = 21 ; CL = D7 ; => DL = 42
ROR EDX,8 ; EDX = 42B2AB59
XOR EDX, 43E2AB9D ; EDX = 015000C4
SHR EAX, 8 ; EAX = 00000088
ROL AL,CL ; AL = 88 ; CL = D7; => AL = 44
SHL EAX, 2; EAX = 00000044 * 4 = 00000110
ADD EAX, DWORD PTR DS:[ (+14) ] ; EAX = 00000110 + 020A7000 = 020A7110
MOV DWORD PTR DS:[EAX], EDX ; EAX = 020A7110 ; EDX = 015000C4
POP EDX ; EDX = 889AFC25
SHL EDX,30 ; EDX = FC250000
SHR EDX,18 ; EDX = 000000FC
ADD DL,CL ; DL = FC ; CL = D7 ; => DL = D3
SHL EDX, 2 ; EDX = 000000D3 * 4 = 0000034C
MOV EAX,DWORD PTR DS:[ (+0) ] ; EAX = FFF2C000
ADD EAX,DWORD PTR DS:[ (+C) ] ; EAX = FFF2C000 + 020A5000 = 01FD1000
ADD EAX,EDX ; EAX = 01FD1000 + 0000034C = 01FD134C
ADD DWORD PTR DS:[ (+4) ],8
MOV EDX, DWORD PTR DS:[EAX] ; EDX = C003FFCE
MOV EAX, DWORD PTR DS:[ (+C) ] ; EAX = 020A5000
MOV EBX, DWORD PTR DS:[ (+14) ]; EBX = 020A7000
PUSH EDX
PUSH EAX
CPUID
```

```
AND EAX,FFFFFFDF
MOV CL,AL ; CL = AL = 92
ROR DWORD PTR SS:[ESP+4],CL ; [ESP+4] = C003FFCE / F0 = FFF3B000
POP EAX ; EAX = 020A5000
ADD DWORD PTR SS:[ESP],EAX ; [ESP] = FFF3B000 + 020A5000 = 01FE0000
POP EAX ; EAX = 01FE0000
JMP EAX

      01FE0000
MOV EAX,DWORD PTR DS:[ (+4) ] ; EAX = FF4B242C
ADD EAX,DWORD PTR DS:[ (+C) ] ; EAX = FF4B242C + 020A5000 = 0155742C
MOV EDX, EAX ;
ADD EDX, 4 ; EDX = 0155742C + 4 = 01557430
MOV EDX,DWORD PTR DS:[EDX] ; EDX = 243BBBD6
MOV EAX,DWORD PTR DS:[EAX] ; EAX = 4A191A68
MOV CL,BYTE PTR DS:[ (+10) ] ; CL = FC
PUSH EAX
SHL EAX,18 ; EAX = 68000000
SHR EAX,18 ; EAX = 00000068
ADD BYTE PTR DS:[ (+10) ],AL ; (+10) = 68 + FC = 64
POP EAX ; EAX = 4A191A68
ROR DL,CL ; DL = D6 ; CL = FC; => DL = 6D
ROR EDX,8 ; EDX = 6D243BBB
ROR DL,CL ; DL = BB ; CL = FC ; => DL = BB
ROR EDX,8 ; EDX = BB6D243B
ROR DL,CL ; DL = 3B ; CL = FC ; => DL = B3
ROR EDX,8 ; EDX = B3BB6D24
ROR DL,CL ; DL = 24 ; CL = FC ; => DL = 42
ROR EDX,8 ; EDX = 42B3BB6D
XOR EDX, 43E2AB9D ; EDX = 015110F0
SHR EAX, 18 ; EAX = 0000004A
ROL AL,CL ; AL = 4A ; CL = FC ; AL = A4
SHL EAX, 2 ; EAX = A4 * 4 = 00000290
ADD EAX,EBX ; EAX = 00000290 + 020A7000 = 020A7290
MOV DWORD PTR DS:[EAX],EDX ;
POP EDX ; EDX = 4A191A68
SHL EDX,30 ; EDX = 1A680000
SHR EDX,18 ; EDX = 0000001A
ADD DL,CL ; DL = 1A + FC = 16
MOV EAX,DWORD PTR DS:[ (+0) ] ; EAX = FFF2C000
ADD EAX,DWORD PTR DS:[ (+C) ] ; EAX = FFF2C000 + 020A5000 = 01FD1000
ADD DWORD PTR DS:[ (+4) ],8 ; (+4) = FF4B242C + 8 = FF4B2434
SHL EDX,2 ; EDX = 0000001A * 4 = 00000058
ADD EDX,EAX ; EDX = 00000058 + 01FD1000 = 01FD1058
MOV EAX,DWORD PTR DS:[ (+C) ] ; EAX = 020A5000
MOV EDX,DWORD PTR DS:[EDX] ; EDX = 8ADC02A6
PUSH EAX
CPUID
AND EAX,FFFFFFDF
MOV CL,AL ; CL = AL = 92
ROR DWORD PTR SS:[ESP+4],CL ; [ESP+4] = 8ADC02A6/ F0 = 00A9A2B7
POP EAX ; EAX = 020A5000
ADD DWORD PTR SS:[ESP],EAX ; [ESP] = 00A9A2B7+ 020A5000 = 02B3F2B7
POP EAX ; EAX = 02B3F2B7
JMP EAX

      02B3F2B7
MOV EAX,DWORD PTR DS:[(+4)] ; EAX = FF4B2434
ADD EAX,DWORD PTR DS:[ (+C) ] ;EAX = FF4B2434 + 020A5000 = 01557434
```

```
MOV EAX,DWORD PTR DS:[EAX] ; EAX = 7D44A708
MOV EDX, EAX ; EDX = 7D44A708
MOV CL,BYTE PTR DS:[ (+10) ] ; CL = 64
PUSH ECX
SHR EDX,10 ; EDX = 00007D44
AND EDX,000000FF ; EDX = 00000044
ADD CL,DL ; CL = 64 + 44 = 4C
MOV EDX, EAX ; EDX = 7D44A708
SHR EDX,18 ; EDX = 0000007D
ADD CL,DL ; CL = 7D + 4C = C9
MOV EDX, EAX ; EDX = 7D44A708
SHR EDX,8 ; EDX = 007D44A7
AND EDX,000000FF ; 000000A7
SUB DL,CL ; DL = A7 – C9 = DE (Carry = 1)
ADD BYTE PTR DS:[ (+10) ],DL ; (+10) = 64 + DE = 42
POP ECX ; ECX = 020A7064
MOV EDX, EAX ; EDX = 7D44A708
MOV ESI, EAX ; ESI = 7D44A708
SHR EDX,18 ; EDX = 0000007D
ROR DL,CL ; DL = 7D ; CL = 64 ; => DL = D7
MOV EAX,DWORD PTR DS:[ (+0) ] ; EAX = FFF2C000
ADD EAX,DWORD PTR DS:[ (+C) ] ; EAX = FFF2C000 + 020A5000 = 01FD1000
SHL EDX,2 ; EDX = 000000D7 * 4 = 0000035C
ADD EAX,EDX ; EAX = 01FD1000 + 0000035C = 01FD135C
MOV EAX,DWORD PTR DS:[EAX] ; EAX = 0003FFEA
PUSH EAX
PUSH ECX
CPUID
AND EAX,FFFFFFDF
MOV CL,AL ; CL = AL = 92
ROR DWORD PTR SS:[ESP+4],CL ; [ESP+4] = 0003FFEA / F0 = FFFA8000
POP ECX ; ECX = 020A7064
MOV EAX,DWORD PTR DS:[ (+C) ] ; EAX = 020A5000
ADD DWORD PTR SS:[ESP],EAX ; [ESP] = FFFA8000 + 020A5000 = 0204D000
MOV EAX, ESI; EAX = 7D44A708
MOV EDX,EAX ; EDX = 7D44A708
AND EDX,000000FF ; EDX = 00000008
SUB DL,CL ; DL = 08 – 64 = A4 (Carry = 1)
SHL EDX,2 ; EDX = 000000A4 * 4 = 00000290
ADD EDX,EBX ; EDX = 00000290 + 020A7000 = 020A7290
PUSH DWORD PTR DS:[EDX] ; [ESP] = 015110F0
MOV EDX,EAX ; EDX = 7D44A708
ROL EDX,10 ; EDX = A7087D44
AND EDX,000000FF ; EDX = 00000044
ROL DL,CL ; DL = 64 ; CL = 44 ; => DL = 44
SHL EDX,2 ; EDX = 00000044 * 4 = 00000110
ADD EDX,EBX ; EDX = 00000110 + 020A7000 = 020A7110
MOV EDX,DWORD PTR DS:[EDX] ; EDX = 015000C4
POP DWORD PTR DS:[EDX] ; [ESP] = 015110F0; EDX = 015000C4
ADD DWORD PTR DS:[ (+4) ],4 ; (+4) = FF4B2434 + 4 = FF4B2438
POP EAX ; EAX = 0204D000
JMP EAX ;

      0204D000
MOV ESI, DWORD PTR DS:[ (+4) ] ; ESI = FF4B2438
ADD ESI,DWORD PTR DS:[ (+C) ] ; ESI = FF4B2438 + 020A5000 = 01557438
PUSH DWORD PTR DS:[ESI] ; [ESP] = 848AB4E3
MOV CL,BYTE PTR DS:[ (+10) ] ; CL = 42
POP EAX; EAX = 848AB4E3
```

```
MOV ESI, EAX; ESI = 848AB4E3
SHR EAX,18 ;  EAX = 00000084
XOR AL, 90 ; AL = 90
ADD BYTE PTR DS:[ (+10) ],AL ; (+10) = 42 + 90 = D2
MOV EAX, ESI; EAX = 848AB4E3
SHL EAX,8 ; EAX = 8AB4E300
SHR EAX,18 ; EAX = 0000008A
ROL AL,CL ; AL = 8A ; CL = 42 ; => AL = 2A
XOR AL,2D ; AL = 07
SHL EAX,2 ; EAX = 00000007 * 4 = 0000001C
ADD EAX, DWORD PTR DS:[ (+0) ]; EAX = 0000001C  + 020A7000  = 020A701C
MOV DWORD PTR DS:[EAX], EAX ; EAX = 0022FF6C
PUSH EAX ;
MOV EAX, ESI ; EAX = 848AB4E3
SHL EAX,18 ; EAX = E3000000
SHR EAX,18 ; EAX = 000000E3
SUB AL,CL ; AL = E3 – 42 = A1
XOR AL,89 ; AL = 28
SHL EAX,2 ; EAX = 00000028 * 4 = 000000A0
ADD EAX,DWORD PTR SS:[ (+0) ] ; EAX = 000000A0 + 020A7000 = 020A70A0
XCNH EAX, EDI ; => EAX = 0022FF6C ; EDI = 020A70A0
POP DWORD PTR DS:[EDI] ; [ESP] = 0022FF6C; EDI = 020A70A0
MOV EAX, ESI ; EAX = 848AB4E3
SHL EAX,10 ; EAX = B4E30000
SHR EAX,18 ; EAX = 000000B4
ROR AL,CL ; AL = B4 ; CL = 42 ; => AL = 2D (/4)
XOR AL,4B ; AL = 66
SHL EAX,2 ; EAX = 00000066 * 4 = 00000198
PUSH ECX ; ECX = 00802092
MOV EDI, DWORD PTR DS:[ (+0) ] ;  EDI = FFF2C000
ADD EDI,DWORD PTR DS:[ (+C) ] ; EDI = FFF2C000 + 020A5000 = 01FD1000
ADD EDI, EAX ; EDI = 01FD1000 + 00000198 = 01FD1198
MOV EDI, DWORD PTR DS:[EDI] ; EDI = C003FFF7
ADD DWORD PTR DS:[ (+4) ],4 ; (+4) = FF4B2438 + 4 = FF4B243C
CPUID
AND EAX,FFFFFFDF
ROR EDI, CL ; EDI = C003FFF7/ F0 = FFFDF000
ADD EDI, DWORD PTR DS:[ (+C) ] ; EDI = FFFDF000 + 020A5000 = 02084000
MOV EBX, DWORD PTR DS:[ (+0) ] ; ECX = 020A7000
POP ECX ; ECX = 00802092
JMP EDI ; EDI = 02084000

      02084000
MOV ESI, DWORD PTR DS:[(+4) ] ; ESI = FF4B243C
ADD ESI,DWORD PTR DS:[ (+C) ] ; ESI = FF4B243C  + 20A5000 = 0155743C
ADD ESI,4 ; ESI  = 0155743C + 4 = 01557440
MOV EDI, DWORD PTR DS:[ESI] ; EDI = 02482334
SUB ESI, 4 ; ESI = 0155743C
MOV ESI, DWORD PTR DS:[ESI] ; ESI = BB6529A2
MOV CL,BYTE PTR DS:[ (+10) ] ; CL = D2
MOV EAX, ESI ; EAX = BB6529A2
SHL EAX,18 ; EAX = A2000000
SHR EAX,18 ; EAX = 000000A2
XOR AL,0E ; EAX = 000000AC
ADD BYTE PTR DS:[ (+10) ],AL ; (+10) = D2 + AC = 7E
MOV EAX, ESI ; EAX = BB6529A2
SHL EAX,8 ; EAX = 6529A200
SHR EAX,18 ; EAX = 00000065
SUB AL,CL ; AL = 65 – D2 = 93
```

```
XOR AL,BB ; AL = 93; => AL = 28
SHL EAX,2 ; EAX = 00000028 * 4 = 000000A0
ADD EAX, EBX ; EAX = 000000A0 + 020A7000 = 020A70A0
MOV EDI, EAX ;
XOR EAX, 02482310h ; EAX = 00000024
MOV EDX, DWORD PTR DS:[ (+8) ] ; EDX = 286
PUSH EDX ;
POPFD ;
ADD DWORD PTR DS:[EDI],EAX ; EDI = 020A70A0 ; DWORD = 0022FF6C + 24 = 0022FF90
PUSHFD ; EFL = 216
POP DWORD PTR DS:[ (+8) ] ; (+8) = 216
MOV ESI, EAX ; EAX = BB6529A2
SHR EAX,18 ; EAX = 000000BB
ROL AL,CL ; AL = BB ; CL = D2 ; => AL = EE
XOR AL,28 ; AL = C6
SHL EAX,2 ; EAX = 000000C6 * 4 = 00000318
MOV EDI, DWORD PTR DS:[ (+0) ] ; EDI = FFF2C000
ADD EDI,DWORD PTR SS:[ (+C) ] ; EDI = FFF2C000 + 020A5000 = 01FD1000
ADD EDI, EAX ; EDI = 01FD1000 + 00000318 = 01FD1318
MOV EDI, DWORD PTR DS:[EDI] ; EDI = 4003FFDF
ADD DWORD PTR DS:[ (+4) ],8 ; (+4) = FF4B243C + 8 = FF4B2444
CPUID
AND EAX,FFFFFFDF
MOV CL,AL ; CL = AL = 92
ROR EDI, CL ; EDI = 4003FFDF / F0 = FFF7D000
ADD EDI, DWORD PTR DS:[ (+C) ] ; EDI = FFF7D000 + 020A5000 = 02022000
JMP EDI ; EDI = 02022000

      02022000
MOV EAX, DWORD PTR DS:[ (+4) ] ; EAX = FF4B2444
ADD EAX,DWORD PTR DS:[ (+C) ] ; EAX = FF4B2444 + 020A5000 = 01557444
MOV EDX, 4;
ADD EDX, EAX ; EDX = 00000004 + 01557444 = 01557448
MOV EDX, DWORD PTR DS:[EDX] ; EDX = 90ECEE5B
MOV EAX, DWORD PTR DS:[EAX] ; EAX = 92CDA6B0
PUSH EAX;
MOV CL,BYTE PTR DS:[ (+10) ] ; CL = 7E
SHL EAX,18 ; EAX = B0000000
SHR EAX,18 ; EAX = 000000B0
ADD BYTE PTR DS:[ (+10) ],AL ; (+10) = 7E + B0 = 2E
POP EAX; EAX = 92CDA6B0
ROR DL,CL ; DL = 5B; CL = 7E; => DL = 6D
ROR EDX,8; EDX = 6D90ECEE
ROR DL,CL ; DL = EE; CL = 7E; => DL = BB
ROR EDX,8 ; EDX = BB6D90EC
ROR DL,CL ; DL = EC; CL = 7E; => DL = B3
ROR EDX,8 ; EDX = B3BB6D90
ROR DL,CL ; DL = 90; CL = 7E; => DL = 42
ROR EDX,8 ; EDX = 42B3BB6D
XOR EDX, 43E2AB9D ; EDX = 015110F0
PUSH EAX ; EAX = 92CDA6B0
SHR EAX, 18 ; EAX = 00000092
ROL AL,CL ; AL = 92; CL = 7E; => AL = A4
SHL EAX,2 ; EAX = 000000A4 * 4 = 00000290
ADD EAX, DWORD PTR DS:[ (+0) ] ; EAX =     00000290 + 020A7000 = 020A7290
MOV DWORD PTR DS:[EAX], EDX ; EDX = 015110F0
POP EDX ; EDX = 92CDA6B0
SHL EDX, 30 ; EDX = A6B00000
SHR EDX, 18 ; EDX = 000000A6
```

```
ADD DL,CL ; DL = A6 + 7E = 24
MOV EAX, DWORD PTR DS:[ (+0) ] ; EAX = FFF2C000
ADD EAX, DWORD PTR DS:[ (+C) ] ; EAX = FFF2C000 + 020A5000 = 01FD1000
ADD DWORD PTR DS:[ (+4) ],8 ; (+4) = FF4B2444 + 8 = FF4B244C
SHL EDX, 2 ; EDX = 00000024 * 4 = 00000090
ADD EDX, EAX ; EDX = 00000090 + 01FD1000 = 01FD1090
MOV EAX , DWORD PTR DS:[ (+C) ] ; EAX = 020A5000
PUSH EAX ;
MOV EDX, DWORD PTR DS:[EDX] ; EDX = 0003FFE0
CPUID
AND EAX,FFFFFFDF
MOV CL,AL ; CL = AL = 92
ROR EDX,CL ; EDI = 0003FFE0/ F0 = FFF80000
POP EAX ; EAX = 020A5000
ADD EDX ,EAX ; EDX = FFF80000 + 020A5000 = 02025000
MOV EDX, EAX ; EAX = 02025000
JMP EAX

      02025000
MOV EAX,DWORD PTR DS:[ (+4) ] ; EAX = FF4B244C
ADD EAX,DWORD PTR DS:[ (+C) ] ; EAX = FF4B244C + 020A5000 = 0155744C
MOV EAX,DWORD PTR DS:[EAX] ; EAX = A029886D
MOV EDX, EAX ; EDX = A029886D
PUSH EDX ;
MOV CL,BYTE PTR DS:[ (+10) ] ; CL = 2E
SHL EAX, 8 ; EAX = 29886D00
SHR EAX, 18 ; EAX = 00000029
ROR AL,CL ; AL = 29 ; CL = 2E ; => AL = A4
SHL EAX,2 ; EAX = 000000A4 * 4 = 00000290
ADD EAX, DWORD PTR DS:[ (+0) ] ; EAX = 00000290 + 020A7000 = 020A7290
MOV EAX, DWORD PTR DS:[EAX] ; EAX = 015110F0
SHR EDX, 18 ; EDX = 000000A0
ROL DL,CL ; DL = A0 ; CL = 2E ; => DL = 28
SHL EDX,2 ; EDX = 00000028 * 4 = 000000A0
ADD EDX,DWORD PTR DS:[ (+0) ] ; EDX = 000000A0 + 020A7000 = 020A70A0
MOV EDX, DWORD PTR DS:[EDX] ; EDX = 0022FF90
MOV DWORD PTR DS:[EDX], EAX ; EAX = 015110F0
POP EDX ; EDX = A029886D
PUSH EDX ;
SHL EDX,18 ; EDX = 6D000000
SHR EDX,18 ; EDX = 0000006D
ADD BYTE PTR DS:[ (+10) ],DL ; (+10) = 2E + 6D = 9B
POP EDX ;
SHL EDX,10 ; EDX = 886D0000
SHR EDX,18 ; EDX = 00000088
ADD DL,CL ; DL =88 + 2E = B6
MOV EAX, DWORD PTR DS:[ (+0) ] ; EAX = FFF2C000
ADD EAX,DWORD PTR DS:[ (+C) ] ; EAX = FFF2C000 + 020A5000 = 01FD1000
ADD DWORD PTR DS:[ (+4) ],4 ; (+4) = FF4B244C + 4 = FF4B2450
SHL EDX,2 ; EDX = 000000B6 * 4 = 000002D8
ADD EDX,EAX ; EDX = 000002D8 + 01FD1000 = 01FD12D8
MOV EAX, DWORD PTR DS:[ (+C) ] ; EAX = 020A5000
PUSH EAX ;
MOV EDX, DWORD PTR DS:[EDX] ; EDX = 02CC02A6
CPUID
AND EAX,FFFFFFDF
MOV CL,AL; CL = AL = 92
ROR EDX,CL ; EDX = 02CC02A6/ F0 = 00A980B3
POP EAX ; EAX = 020A5000
```

```
ADD EDX, EAX ; EDX = 00A980B3 + 020A5000 = 02B3D0B3
MOV EAX, EDX ; EAX = 02B3D0B3
JMP EAX

      02B3D0B3
MOV DWORD PTR DS:[ (+24) ],0 ;
POPFD ;
POPAD ;
LEA ESP,[ESP+30] ;
RETN 4 ; [ESP] = 015110F0


--------------------------------------------------------------------------
Способ №1A.
0044F669 CALL EAX
База запроса(по стеку):
0022FCF4  004011C3
0022FCF8  00406E34
Обвертка виртуального стека для вызова:
004011C3
Верхушка виртуального стека:
014FACE1
Окончание виртуального стека:
014FAD25
Количество используемых DWORDs в виртуальном стеке:
16
Адрес запрашиваемой WinAPI:
SetUnhandledExceptionFilter
--------------------------------------------------------------------------
            02B3B7CE
            2054000
MOV ESI,DWORD PTR DS:[ (+4) ] ; ESI = FF455CF1
ADD ESI,DWORD PTR DS:[ (+C) ] ; ESI = FF455CF1 + 020A5000 = 014FACF1
MOV ESI, DWORD PTR DS:[ESI] ; ESI = 6E895B39
MOV CL,BYTE PTR DS:[ (+10) ] ; CL = 67
MOV EAX, ESI ; EAX = 6E895B39
SHL EAX,10 ; EAX = 5B390000
SHR EAX,18 ; EAX = 0000005B
XOR AL,07; AL = 5C
ADD BYTE PTR DS:[ (+10) ],AL ; (+10) = 67 + 5C = C3
MOV EAX, ESI ; EAX = 6E895B39
SHL EAX,18 ; EAX = 39000000
SHR EAX,18 ; EAX = 00000039
ROR AL,CL ; AL = 39 ; CL = 67 ; => AL = 72
XOR AL,5B ; EAX = 00000029
SHL EAX,2 ; EAX = 00000029 * 4 = 000000A4
ADD EAX, DWORD PTR DS:[ (+14) ] ; EAX = 000000A4 + 020A7000 = 020A70A4
MOV EAX, DWORD PTR DS:[EAX] ; EAX = 0022FCF0
MOV EAX, DWORD PTR DS:[EAX] ; EAX = 0044F66B
PUSH EAX
MOV EAX, ESI ; EAX = 6E895B39
SHR EAX,18 ; EAX = 0000006E
SUB AL,CL ; AL = 6E − 67 = 07
XOR AL,2E ; AL = 29
SHL EAX,2 ; EAX = 00000029 * 4 = 000000A4
ADD DWORD PTR SS:[ESP],EBX ;
ADD EAX, DWORD PTR DS:[ (+14) ] ; EAX = 000000A4 + 020A7000 = 020A70A4
MOV EDI, EAX ; EDI = 020A70A4
POP DWORD PTR DS:[EDI] ; [ESP] = 0044F66B
```

```
MOV EAX, ESI ; EAX = 6E895B39
SHL EAX,8 ; EAX = 895B3900
SHR EAX,18 ; EAX = 00000089
ADD AL,CL ; AL = 89 + 67 = F0
XOR AL,AA ; AL = 5A
SHL EAX,2 ; EAX = 0000005A * 4 = 00000168
MOV EDI, DWORD PTR DS:[ (+0) ] ; EDI = FFF2C000
ADD EDI, DWORD PTR DS:[ (+C) ] ; EDI = FFF2C000 + 020A5000 = 01FD1000
ADD EDI, EAX ; EDI = 01FD1000 + 00000168 = 01FD1168
MOV EDI, DWORD PTR DS:[EDI] ; EDI = 9F3802A5
ADD DWORD PTR DS:[ (+4) ], 4 ; (+4) = FF455CF1 + 4 = FF455CF5
CPUID
AND EAX,FFFFFFDF
MOV CL,AL ; CL = AL = 92
ROR EDI,CL ; EDI = 9F3802A5/ F0 = 00A967CE
ADD EDI,DWORD PTR DS:[ (+C) ]; EDX = 00A967CE + 020A5000 = 02B3B7CE
JMP EDI

      02B3B7CE
      02084000
      01FD2000
MOV EAX,DWORD PTR DS:[ (+4) ] ; EAX = FF455D01
ADD EAX,DWORD PTR DS:[ (+C) ] ; EAX = 014FAD01
MOV EAX,DWORD PTR DS:[EAX] ; EAX = C0255B1C
MOV EDX,EAX ; EDX = EAX = C0255B1C
MOV CL,BYTE PTR DS:[ (+10) ] ; CL = 3D
SHL EAX, 8 ; EAX = 255B1C00
SHR EAX, 18 ; EAX = 00000025
ROR AL,CL ; AL = 25 ; CL = 3D ; => AL = 29
SHL EAX,2 ; EAX = 00000029 * 4 = 000000A4
ADD EAX, DWORD PTR DS:[ (+14) ] ; EAX = 000000A4 + 020A7000 = 020A70A4
MOV EAX,DWORD PTR DS:[EAX] ; EAX = 0044F66B
PUSH EDX
SHR EDX,18 ; EDX = 000000C0
ROL DL,CL ; DL = C0 ; CL = 3D ; => DL = 18
SHL EDX,2 ; EDX = 00000018 * 4 = 00000060
ADD EDX, DWORD PTR DS:[ (+14) ] ; EDX = 00000060 + 020A7000 = 020A7060
MOV EDX,DWORD PTR DS:[EDX] ; EDX = 0022FCF4
MOV DWORD PTR DS:[EDX],EAX ; EAX = 0044F66B
POP EDX ; EDX = C0255B1C
PUSH EDX ;
SHL EDX,18 ; EDX = 1C000000
SHR EDX,18 ; EDX = 0000001C
ADD BYTE PTR DS:[ (+10) ],DL ; (+10) = 3D + 1C = 59
POP EDX ; EDX = C0255B1C
SHL EDX,10 ; EDX = 5B1C0000
SHR EDX,18 ; EDX = 0000005B
ADD DL,CL ; DL = 5B + 3D = 98
MOV EAX, DWORD PTR DS:[ (+0) ] ; EDI = FFF2C000
ADD EAX, DWORD PTR DS:[ (+C) ] ; EDI = FFF2C000 + 020A5000 = 01FD1000
ADD DWORD PTR DS:[ (+4) ],4 ; (+4) = FF455D01 + 4 = FF455D05
SHL EDX,2 ; EDX = 0000005B * 4 = 00000260
ADD EDX,EAX ; EDX = 00000260 + 01FD1000 = 01FD1260
MOV EDX,DWORD PTR DS:[EDX] ; EDX = BDE802A6
CPUID
AND EAX,FFFFFFDF
MOV CL,AL ; CL = AL = 92
ROR EDX,CL ; EDI = BDE802A6/ F0 = 00A9AF7A
ADD EDX,DWORD PTR DS:[ (+C) ]; EDX = 00A9AF7A + 020A5000 = 02B3FF7A
```

```
MOV EAX, EDX ; EAX = 02B3FF7A
JMP EAX

      02B3FF7A
MOV EAX,DWORD PTR DS:[ (+4) ] ; EAX = FF455D05
ADD EAX,DWORD PTR DS:[ (+C) ] ; EAX = 014FAD05
MOV EAX,DWORD PTR DS:[EAX] ; EAX = 709D7E56
MOV EDX,EAX ; EDX = 709D7E56
PUSH EDX
MOV CL,BYTE PTR DS:[ (+10) ] ; CL = 59
SHL EDX,18 ; EDX = 56000000
SHR EDX,18 ; EDX = 00000056
ADD BYTE PTR DS:[ (+10) ],DL ; (+10) = 59 + 56 = AF
POP EDX ; EDX = 709D7E56
SHL EDX,10 ; EDX = 7E560000
SHR EDX,18 ; EDX = 0000007E
ADD DL,CL ; DL = 7E + 59 = D7
MOV EAX,DWORD PTR DS:[EBX] ; EAX = FFF2C000
ADD EAX,DWORD PTR DS:[ (+C) ] ; EAX = 01FD1000
ADD DWORD PTR DS:[ (+4) ],4 ; (+4) = FF455D05 + 4 = FF455D09
SHL EDX,2 ; EDX = 000000D7 * 4 = 0000035C
ADD EDX,EAX ; EDX = 0000035C + 01FD1000 = 01FD135C
MOV EDX,DWORD PTR DS:[EDX] ; EDX = 0003FFEA
CPUID
AND EAX,FFFFFFDF
MOV CL,AL ; CL = AL = 92
ROR EDX,CL ; EDI = 0003FFEA/ F0 = FFFA8000
ADD EDX,DWORD PTR DS:[ (+C) ]; EDX = FFFA8000 + 020A5000 = 0204D000
MOV EAX, EDX ; EAX = 0204D000
JMP EAX

      0204D000
      01FFB000
MOV ESI,DWORD PTR DS:[ (+4) ] ; ESI = FF455D0D
ADD ESI,DWORD PTR DS:[ (+C) ] ; ESI = FF455D0D + 020A5000 = 014FAD0D
ADD ESI, 4 ; ESI = 014FAD11
MOV EDI, DWORD PTR DS:[ESI] ; EDI = 017A0D64
SUB ESI, 4 ; ESI = 014FAD0D
MOV ESI, DWORD PTR DS:[ESI] ; ESI = 615D49DC
MOV CL,BYTE PTR DS:[ (+10) ] ; CL = 1E
MOV EAX, ESI; EAX = 615D49DC
SHL EAX,18 ; EAX = DC000000
SHR EAX,18 ; EAX = 000000DC
XOR AL,0B ; EAX = 000000D7
ADD BYTE PTR DS:[ (+10) ],AL ; (+10) = 1E + D7 = F5
MOV EAX, EDI; EAX = 017A0D64
XOR EAX,017A0D40 ; EAX = 00000024
PUSH EAX
MOV EAX, ESI ; EAX = 615D49DC
SHR EAX,18 ; EAX = 00000061
SUB AL,CL ; AL = 61 – 1E = 43
XOR AL,5B ; AL = 18
SHL EAX,2 ; EAX = 00000018 * 4 = 00000060
ADD EAX,DWORD PTR DS:[ (+14) ] ; EAX = 00000060 + 020A7000 = 020A7060
MOV EDI, EAX; EDI = 020A7060
POP EAX ; EAX = 00000024
MOV EDX, DWORD PTR DS:[ (+8) ] ; EDX = 212
PUSH EDX ;
POPFD ;
```

```
ADD DWORD PTR DS:[EDI],EAX ; EDI = 020A7060; DWORD = 0022FCCC + 24 = 22FCF0
PUSHFD ; EFL = 216
POP DWORD PTR DS:[ (+8) ] ; (+8) = 216
MOV EAX, ESI; EAX = 615D49DC
SHL EAX,10 ; EAX = 49DC0000
SHR EAX,18 ; EAX = 00000049
ROL AL,CL ; AL = 49 ; CL = 1E ; => AL = 52
XOR AL,8F ; AL = DD
SHL EAX,2 ; EAX = 000000DD * 4 = 00000374
MOV EDI, DWORD PTR DS:[ (+0) ] ; EDI = FFF2C000
ADD EDI,DWORD PTR SS:[ (+C) ] ; EDI = FFF2C000 + 020A5000 = 01FD1000
ADD EDI, EAX ; EDI = 01FD1000 + 00000374 = 01FD1374
MOV EDI, DWORD PTR DS:[EDI] ; EDI = 0003FFF7
ADD DWORD PTR DS:[ (+4) ],8 ; (+4) = FF455D0D + 8 = FF455D15
CPUID
AND EAX,FFFFFFDF
MOV CL,AL ; CL = AL = 92
ROR EDI, CL ; EDI = 0003FFF7 / F0 = FFFDC000
ADD EDI, DWORD PTR DS:[ (+C) ] ; EDI = FFFDC000 + 020A5000 = 02081000
JMP EDI ; EDI = 02081000

      02081000
MOV EAX,DWORD PTR DS:[ (+4) ] ; EAX = FF455D15
ADD EAX,DWORD PTR DS:[ (+C) ] ; EAX = 014FAD15
MOV EDX, EAX; EDX = 014FAD15
ADD EDX, 4 ; EDX = 014FAD19
MOV EDX, DWORD PTR DS:[EDX] ; EDX = 68085F2A
MOV EAX,DWORD PTR DS:[EAX] ; EAX = 604C616C
PUSH EAX
MOV CL,BYTE PTR DS:[ (+10) ] ; CL = F5
SHL EAX,18 ; EAX = 6C000000
SHR EAX,18 ; EAX = 0000006C
ADD BYTE PTR DS:[ (+10) ],AL ; (+10) = F5 + 6C = 61
ROR DL,CL ; DL = 2A ; CL = F5 ; => DL = 51
ROR EDX,8; EDX = 5168085F
ROR DL,CL ; DL = 5F; CL = F5; => DL = FA
ROR EDX,8 ; EDX = FA516808
ROR DL,CL ; DL = 08; CL = F5; => DL = 40
ROR EDX,8 ; EDX = 40FA5168
ROR DL,CL ; DL = 68; CL = F5; => DL = 43
ROR EDX,8 ; EDX = 4340FA51
XOR EDX, 43E2AB9D ; EDX = 00A251CC
POP EAX ; EAX = 604C616C
PUSH EAX ;
SHR EAX, 18 ; EAX = 00000060
ROL AL,CL  ; AL = 60 ; CL = F5 ; => AL = 0C
SHL EAX,2 ; EAX = 0000000C * 4 = 00000030
ADD EAX, DWORD PTR DS:[ (+14) ] ; EAX = 00000030 + 020A7000 = 020A7030
MOV DWORD PTR DS:[EAX], EDX ;
POP EAX ; EAX = 604C616C
SHL EDX,30 ; EAX = 616C0000
SHR EDX,18 ; EAX = 00000061
ADD DL,CL ; DL = 61 + F5 = 56
MOV EAX,DWORD PTR DS:[ (+0) ] ; EAX = FFF2C000
ADD EAX,DWORD PTR DS:[ (+C) ] ; EAX = 01FD1000
ADD DWORD PTR DS:[ (+4) ],8 ; (+4) = FF455D15 + 8 = FF455D1D
SHL EDX,2 ; EDX = 00000056 * 4 = 00000158
ADD EDX,EAX ; EDX = 00000158 + 01FD1000 = 01FD1158
MOV EDX,DWORD PTR DS:[EDX] ; EDX = C003FFEB
```

```
CPUID
AND EAX,FFFFFFDF
MOV CL,AL ; CL = AL = 92
ROR EDX,CL ; EDI = C003FFEB / F0 = FFFA8000
ADD EDX,DWORD PTR DS:[ (+C) ]; EDX = FFFA8000 + 020A5000 = 02054000
MOV EAX, EDX ; EAX = 02054000
JMP EAX
```

02054000
02B3B7CE
02084000
01FD2000
02B3FF7A
0204D000
01FFB000
02081000
02054000
01FD2000

Способ №2. Прямой ход алгоритма декодирования 2го аргумента(EBP+0xC) с константами(операндами) по адресу в 1м аргументе(EBP+8)

```
01520C0A    PUSH EBP
```

База запроса:

```
01520C38    PUSH 00401261
01520C3D    PUSH OFFSET 01520C4F
```

Обвертка виртуального стека для вызова:

```
00401261
```

Верхушка виртуального стека:

```
014F954D
```

Окончание виртуального стека:

```
014F9AA9
```

Количество используемых DWORDs в виртуальном стеке:

```
342
```

Адрес возврата:

```
1520C4F
```

Зашитые скрытые асм команды за способом (адрес островка 01FDE005):

```
AND EAX, 5CAC5AC5
AND ECX, 5CAC5AC5
```

+ КОПИРОВАНИЕ ОПЕРАНДОВ ДЛЯ АЛГОРИТМА В СТЕК

02B3B7CE
2033000

```
MOV EAX,DWORD PTR DS:[ (+4) ] ; EAX = FF454555
ADD EAX,DWORD PTR DS:[ (+C) ] ; EAX = 014F9555
MOV EAX,DWORD PTR DS:[EAX] ; EAX = FF8992DC
MOV EDX, EAX ; EDX = FF8992DC
MOV ECX,DWORD PTR DS:[ (+10) ] ; ECX = 0000009C
SHR EDX,18 ; EDX = 000000FF
XOR DL,19 ; DL = E6
ADD BYTE PTR DS:[ (+10) ],DL ; (+10) = 9C + E6 = 82
MOV EDX, EAX ; EDX = FF8992DC
SHL EDX,18 ; EDX = DC000000
SHR EDX,18 ; EDX = 000000DC
ROR DL,CL ; DL = DC; CL = 9C; => DL = CD
XOR DL,CA ; DL = 07
SHL EDX, 2 ; EDX = 00000007 * 4 = 0000001C
```

```
ADD EDX, DWORD PTR DS:[ (+14) ]; EDX = 0000001C + 020A7000 = 20A701C
MOV EDX, DWORD PTR DS:[EDX] ; EDX = 0022F7D4
PUSH EDX
MOV EDX, EAX ; EDX = FF8992DC
SHL EDX,10 ; EDX = 92DC0000
SHR EDX,18 ; EDX = 00000092
ROR DL,CL ; DL = 92; CL = 9C; => DL = 29
XOR DL,3D ; DL = 14
SHL EDX, 2 ; EDX = 00000014 * 4 = 00000050
ADD EDX, DWORD PTR DS:[ (+14) ]; EDX = 00000050 + 020A7000 = 20A7050
POP DWORD PTR DS:[EDX] ; [ESP] = 0022F7D4
ADD DWORD PTR DS:[ (+4) ],4 ; (+4) = FF454555 + 4 = FF454559
MOV EDX, DWORD PTR DS:[ (+0) ] ; EDX = FFF2C000
ADD EDX,DWORD PTR SS:[ (+C) ] ; EDX = FFF2C000 + 020A5000 = 01FD1000
SHL EAX,8 ; EAX = 8992DC00
SHR EAX,18 ; EAX = 00000089
ADD AL,CL ; AL = 89 + 9C = 25
XOR AL,D2 ; AL = F7
SHL EAX, 2 ; EAX = 000000F7 * 4 = 000003DC
ADD EDX, EAX ; EDX = 01FD1000 + 000003DC = 01FD13DC
MOV EDX, DWORD PTR DS:[EDX] ; EDX = C84402A6
CPUID
AND EAX,FFFFFFDF
MOV CL,AL ; CL = AL = 92
ROR EDX, CL ; EDX = C84402A6 / F0 = 00A9B211
ADD EDX, DWORD PTR DS:[ (+C) ] ; EDX = 00A9B211 + 020A5000 = 02B40211
JMP EDX ; EDX = 02B40211
```

02B40211 (ОТКРЫТЫЙ, БЕЗ ОБФУСКАЦИИ)

```
MOV EAX,DWORD PTR DS:[EBX+4]
ADD EAX,DWORD PTR DS:[EBX+0C]
MOV ESI,DWORD PTR DS:[EAX+4]
MOV EAX,DWORD PTR DS:[EAX]
MOV EDX,EAX
MOV CL,BYTE PTR DS:[EBX+10]
AND EDX,000000FF
ADD BYTE PTR DS:[EBX+10],DL
MOV EDX,ESI
ROR DL,CL
ROL EDX,8
ROR DL,CL
ROL EDX,8
ROR DL,CL
ROL EDX,8
ROR DL,CL
ROL EDX,8
MOV ESI,EDX
MOV EDX,EAX
SHR EDX,18
ADD DWORD PTR DS:[EDX*4+EBX],ESI
ADD DWORD PTR DS:[EBX+4],8
MOV EDX,DWORD PTR DS:[EBX]
ADD EDX,DWORD PTR DS:[EBX+0C]
SHL EAX,10
SHR EAX,18
ADD AL,CL
PUSH DWORD PTR DS:[EAX*4+EDX]
MOV EAX,1
PUSH EBX
```

```
CPUID
AND EAX,FFFFFFDF
POP EBX
MOV CL,AL
MOV EAX,DWORD PTR DS:[EBX+0C]
ROR DWORD PTR SS:[ESP],CL
ADD DWORD PTR SS:[ESP],EAX
POP EAX
JMP EAX
```

0208B000

```
MOV EAX,DWORD PTR DS:[ (+4) ] ; EAX = FF454561
ADD EAX,DWORD PTR DS:[ (+C) ] ; EAX = 014F9561
MOV EAX, DWORD PTR DS:[EAX] ;  EAX = F95D981A
MOV EDX, EAX; EDX = F95D981A
MOV ECX,DWORD PTR DS:[ (+10) ] ; ECX = 0000008A
SHL EDX,10 ; EDX = 981A0000
SHR EDX,18 ; EDX = 00000098
XOR DL,42 ; DL = 000000DA
ADD BYTE PTR DS:[ (+10) ],DL ; (+10) = 8A + DA = 64
MOV EDX, EAX; EDX = F95D981A
SHL EDX,8 ; EDX = 5D981A00
SHR EDX,18 ; EDX = 0000005D
ADD DL,CL ; DL = 5D + 8A = E7
XOR DL,E0 ; DL = 7
SHL EDX, 2 ; EDX = 00000007 * 4 = 0000001C
ADD EDX, DWORD PTR DS:[ (+14) ]; EDX = 0000001C + 020A7000 = 20A701C
MOV EDX, DWORD PTR DS:[EDX] ; EDX = 0022F7D4
PUSH EDX ;
MOV EDX, EAX; EDX = F95D981A
SHL EDX,18 ; EDX = 1A000000
SHR EDX,18 ; EDX = 0000001A
ROL DL,CL ; DL = 1A; CL = 8A; => DL = 68
XOR DL,37 ; DL = 5F
SHL EDX, 2 ; EDX = 0000005F * 4 = 000017C
ADD EDX, DWORD PTR DS:[ (+14) ]; EDX = 0000017C + 020A7000 = 20A717C
POP DWORD PTR DS:[EDX] ; [ESP] = 0022F7D4
ADD DWORD PTR DS:[ (+4) ],4 ; (+4) = FF454561 + 4 = FF454565
MOV EDX, DWORD PTR DS:[ (+0) ] ; EDX = FFF2C000
ADD EDX,DWORD PTR SS:[ (+C) ] ; EDX = FFF2C000 + 020A5000 = 01FD1000
SHR EAX,18 ; EAX = 000000F9
ROL AL,CL ; AL = F9; CL = 8A; => AL = E7
XOR AL,10 ; AL = F7
SHL EAX, 2 ; EAX = 000000F7 * 4 = 000003DC
ADD EDX, EAX ; EDX = 01FD1000 + 000003DC = 01FD13DC
MOV EDX, DWORD PTR DS:[EDX] ; EDX = C84402A6
CPUID
AND EAX,FFFFFFDF
MOV CL,AL ; CL = AL = 92
ROR EDX, CL ; EDX = C84402A6 / F0 = 00A9B211
ADD EDX, DWORD PTR DS:[ (+C) ] ; EDX = 00A9B211 + 020A5000 = 02B40211
JMP EDX ; EDX = 02B40211
… (ЦИКЛ ПО КОПИРОВАНИЮ ОПЕРАНДОВ ИЗ 00B93AFC В СТЕК)
```

01FDE005 (AND ARG.2)

```
MOV EAX,DWORD PTR DS:[ (+4) ] ; EAX = FF454A4D
ADD EAX,DWORD PTR DS:[ (+C) ] ; EAX = 014F9A4D
MOV EAX, DWORD PTR DS:[EAX] ;  EAX = 15AC1F70
```

```
MOV EDX,EAX ; EDX = 15AC1F70
PUSH EDX ;
MOV CL,BYTE PTR DS:[ (+10) ] ; CL = 11
SHL EAX,8; EAX = AC1F7000
SHR EAX,18; EAX = 000000AC
ROR AL,CL ; AL = AC; CL = 11; => AL = 56
SHL EAX,2 ; EAX = 00000056 * 4 = 00000158
ADD EAX, DWORD PTR DS:[(+14)]; EAX = 00000158 + 020A7000 = 020A7158
MOV EAX,DWORD PTR DS:[EAX] ; EAX = 5CAC5AC5
SHR EDX,18 ; EDX = 00000015
ROL DL,CL ; DL = 15; CL = 11; => DL = 2A
SHL EDX,2 ; EDX = 0000002A * 4 = 000000A8
ADD EDX,DWORD PTR DS:[(+14)]; EDX = 000000A8 + 020A7000  = 020A70A8
MOV EDX,DWORD PTR DS:[EDX] ; EDX = 0022F7F4
AND DWORD PTR DS:[EDX],EAX ; [EDX] = 0000000
POP EDX ; EDX = 15AC1F70
PUSH EDX;
SHL EDX,18; EDX = 70000000
SHR EDX,18; EDX = 00000070
ADD BYTE PTR DS:[(+10)],DL ; (+10) = 11 + 70 = 81
POP EDX ; EDX = 15AC1F70
SHL EDX,10 ; EDX = 1F700000
SHR EDX,18 ; EDX = 0000001F
ADD DL,CL ; DL = 1F + 11 = 30
MOV EAX, DWORD PTR DS:[ (+0) ] ; EAX = FFF2C000
ADD EAX,DWORD PTR SS:[ (+C) ] ; EAX = FFF2C000 + 020A5000 = 01FD1000
ADD DWORD PTR DS:[(+4)],4 ; (+4) = FF454A4D + 4 = FF454A51
SHL EDX, 2 ; EDX = 0000001F * 4 = 000000C0
ADD EDX, EAX ; EAX = 000000C0 + 01FD1000 = 01FD10C0
MOV EDX, DWORD PTR DS:[EDX] ; EAX = C003FFF2
CPUID
AND EAX,FFFFFFDF
MOV CL,AL ; CL = AL = 92
ROR EDX, CL ; EDX = C003FFF2/ F0 = FFFCB000
ADD EDX, DWORD PTR DS:[ (+C) ] ; EDX = FFFCB000 + 020A5000 = 02070000
MOV EAX, EDX ; EAX = 02070000
JMP EAX
```

ПРИВЕДЕННЫЙ ВИД АЛГОРИТМА РАБОТЫ С АРГУМЕНТОМ – ПРЯМОЙ ХОД

Способ №2 - 01520C38

!ДЛЯ СЛУЧАЯ В 1М АРГУМЕНТЕ КОГДА АДРЕС ДЛЯ СЧИТЫВАНИЯ ОПЕРАНДОВ - 00B93AFC

```
-- СКРЫТЫЕ АСМ КОМАНДЫ В СПОСОБЕ №2 ПРИ ПРЯМОМ ХОДЕ АЛГОРИТМА
MOV ECX, DWORD PTR SS:[EBP+8] //00B93AFC
MOV ECX, DWORD PTR DS:[ECX] //5CAC5AC5
MOV EAX, DWORD PTR SS:[EBP+0C] //значение для декодирования
AND EAX, ECX //AND EAX, 5CAC5AC5
MOV ECX, EAX
----------

ADD ECX, 2371E3A6
IMUL ECX, 6ECEF435
XOR EAX,ECX
MOV ECX,DWORD PTR SS:[EBP+0C]
AND ECX, 5CAC5AC5
```

```
ADD ECX, CC45E78D
IMUL ECX, 32E6B04C
XOR EAX,ECX
AND EAX, A353A53A
XOR EAX,DWORD PTR SS:[EBP+0C]
MOV ECX,DWORD PTR SS:[EBP+0C]
AND ECX, 5CAC5AC5
MOV EDX,DWORD PTR SS:[EBP+0C]
AND EDX, 5CAC5AC5
ADD EDX, 2371E3A6
IMUL EDX, 6ECEF435
XOR ECX,EDX
MOV EDX,DWORD PTR SS:[EBP+0C]
AND EDX, 5CAC5AC5
ADD EDX, CC45E78D
IMUL EDX, 32E6B04C
XOR ECX,EDX
AND ECX, A353A53A
XOR ECX,DWORD PTR SS:[EBP+0C]
AND ECX, A353A53A
MOV EDX,DWORD PTR SS:[EBP+0C]
AND EDX, 5CAC5AC5
MOV ESI,DWORD PTR SS:[EBP+0C]
AND ESI, 5CAC5AC5
ADD ESI, 2371E3A6
IMUL ESI, 6ECEF435
XOR EDX,ESI
MOV ESI,DWORD PTR SS:[EBP+0C]
AND ESI, 5CAC5AC5
ADD ESI, CC45E78D
IMUL ESI, 32E6B04C
XOR EDX,ESI
AND EDX, A353A53A
XOR EDX,DWORD PTR SS:[EBP+0C]
AND EDX, A353A53A
ADD EDX, 4BFD7FCD
IMUL EDX, 29FDDBF3
XOR ECX,EDX
MOV EDX,DWORD PTR SS:[EBP+0C]
AND EDX, 5CAC5AC5
MOV ESI,DWORD PTR SS:[EBP+0C]
AND ESI, 5CAC5AC5
ADD ESI, 2371E3A6
IMUL ESI, 6ECEF435
XOR EDX,ESI
MOV ESI,DWORD PTR SS:[EBP+0C]
AND ESI, 5CAC5AC5
ADD ESI, CC45E78D
IMUL ESI, 32E6B04C
XOR EDX,ESI
AND EDX, A353A53A
XOR EDX,DWORD PTR SS:[EBP+0C]
AND EDX, A353A53A
ADD EDX, E1B3EE35
IMUL EDX, 0742F06E
XOR ECX,EDX
AND ECX, 5CAC5AC5
XOR EAX,ECX
```

ПРИВЕДЕННЫЙ ВИД АЛГОРИТМА РАБОТЫ С АРГУМЕНТОМ – ОБРАТНЫЙ ХОД

Способ №2 (ТОЛЬКО КОПИРУЕТ ОПЕРАНДЫ В СТЕК)

```
0152256B    PUSH 00401283
01522570    PUSH OFFSET 0152257E
01522575    JMP 00D44F40

MOV EAX, A353A53A
AND EAX,DWORD PTR SS:[EBP+0C]
MOV ECX, A353A53A
AND ECX,DWORD PTR SS:[EBP+0C]
ADD ECX, 4BFD7FCD
IMUL ECX, 29FDDBF3
XOR EAX,ECX
MOV ECX, A353A53A
AND ECX,DWORD PTR SS:[EBP+0C]
ADD ECX, E1B3EE35
IMUL ECX, 0742F06E
XOR EAX,ECX
AND EAX, 5CAC5AC5
XOR EAX,DWORD PTR SS:[EBP+0C]
MOV ECX, A353A53A
AND ECX,DWORD PTR SS:[EBP+0C]
MOV EDX, A353A53A
AND EDX,DWORD PTR SS:[EBP+0C]
ADD EDX, 4BFD7FCD
IMUL EDX, 29FDDBF3
XOR ECX,EDX
MOV EDX, A353A53A
AND EDX,DWORD PTR SS:[EBP+0C]
ADD EDX, E1B3EE35
IMUL EDX, 0742F06E
XOR ECX,EDX
AND ECX, 5CAC5AC5
XOR ECX,DWORD PTR SS:[EBP+0C]
AND ECX, 5CAC5AC5
MOV EDX, A353A53A
AND EDX,DWORD PTR SS:[EBP+0C]
MOV ESI, A353A53A
AND ESI,DWORD PTR SS:[EBP+0C]
ADD ESI, 4BFD7FCD
IMUL ESI, 29FDDBF3
XOR EDX,ESI
MOV ESI, A353A53A
AND ESI,DWORD PTR SS:[EBP+0C]
ADD ESI, E1B3EE35
IMUL ESI, 0742F06E
XOR EDX,ESI
AND EDX, 5CAC5AC5
XOR EDX,DWORD PTR SS:[EBP+0C]
AND EDX, 5CAC5AC5
ADD EDX, 2371E3A6
IMUL EDX, 6ECEF435
XOR ECX,EDX
MOV EDX, A353A53A
AND EDX,DWORD PTR SS:[EBP+0C]
MOV ESI, A353A53A
AND ESI,DWORD PTR SS:[EBP+0C]
ADD ESI, 4BFD7FCD
IMUL ESI, 29FDDBF3
```

```
XOR EDX,ESI
MOV ESI, A353A53A
AND ESI,DWORD PTR SS:[EBP+0C]
ADD ESI, E1B3EE35
IMUL ESI, 0742F06E
XOR EDX,ESI
AND EDX, 5CAC5AC5
XOR EDX,DWORD PTR SS:[EBP+0C]
AND EDX, 5CAC5AC5
ADD EDX, CC45E78D
IMUL EDX, 32E6B04C
XOR ECX,EDX
AND ECX, A353A53A
XOR EAX,ECX
```

ТАБЛИЦА ПОСТАНОВКИ СПОСОБОМ №2 КОСНТАНТ В СТЕК(ПРЯМОЙ ХОД И ОБРАТНЫЙ ХОД)*

ОПЕРАНДЫ	**СМЕЩЕНИЕ ОТ ЗАЯВЛЕННОГО В АРГУМЕНТЕ 1 АДРЕСА** (00B93AFC)	**МЕСТО В СТЕКЕ**
6ECEF435	+2C	[EBP-28]
CC45E78D	+30	[EBP-24]
5CAC5AC5	+13	[EBP-20]
29FDDBF3	+18	[EBP-1C]
4BFD7FCD	+1C	[EBP-18]
0742F06E	+20	[EBP-14]
E1B3EE35	+24	[EBP-C]
32E6B04C	+34	[EBP-8]
2371E3A6	+28	[EBP-4]

* Адрес [EBP-10] закреплен за результатом, значение будет положено в конце алгоритма.

ТАБЛИЦА ОСНОВНЫХ ПЕРЕМЕННЫХ ЗАДАВАЕМЫХ ВО 2М АРГУМЕНТЕ(АДРЕС КОНСТАНТ 00B93AFC**)**

Прямой ход – Арумент/ Обратный ход- Результат	**Прямой ход- результат/Обратный ход - аргумент**	**Прямой ход: AND EAX, 5CAC5AC5**	**Прямой ход: AND EAX, A353A53A***
0	3DB1F4C7	0	0
A590217B	0	04800041	A110213A
0790A442	1	04800040	0310A402
630A0CFD	2	400808C5	23020438
615BA9FC	3	400808C4	2143A138
2783017F	4	04800045	2303013A
A5D1A57E	5	04800044	0151A53A
401AA8F9	6	400808C1	0012A038
E0092DC0	7	400808C0	A0012500

*NOT(5CAC5AC5) = A353A53A
(XOR) 04800041 ^ A110213A = A590217B

ЭКВИВАЛЕНТНЫЙ АСМ КОД С УСЛОВНЫМИ ПЕРЕХОДАМИ
(СПРАВЕДЛИВО ТОЛЬКО ДЛЯ ЧИСЕЛ В ВЫШЕПРИВЕДЕННОЙ ТАБЛИЦЕ):

```
__asm
{
MOV EAX, Init
AND EAX, 0x5CAC5AC5
CMP EAX, 0x40000000
JG HI
JP LOW_EVEN
SUB EAX, 0x04800041
JMP XI
LOW_EVEN:
SUB EAX, 0x04800040
ADD EAX, 1
JMP XI
HI:
JNP HI_EVEN
MOV ECX, 0x400808C5
SUB ECX, EAX
ADD ECX, 2
MOV EAX, ECX
JMP XI
HI_EVEN:
MOV ECX, 0x400808C4
SUB ECX, EAX
ADD ECX, 3
MOV EAX, ECX
JMP XI
}
```