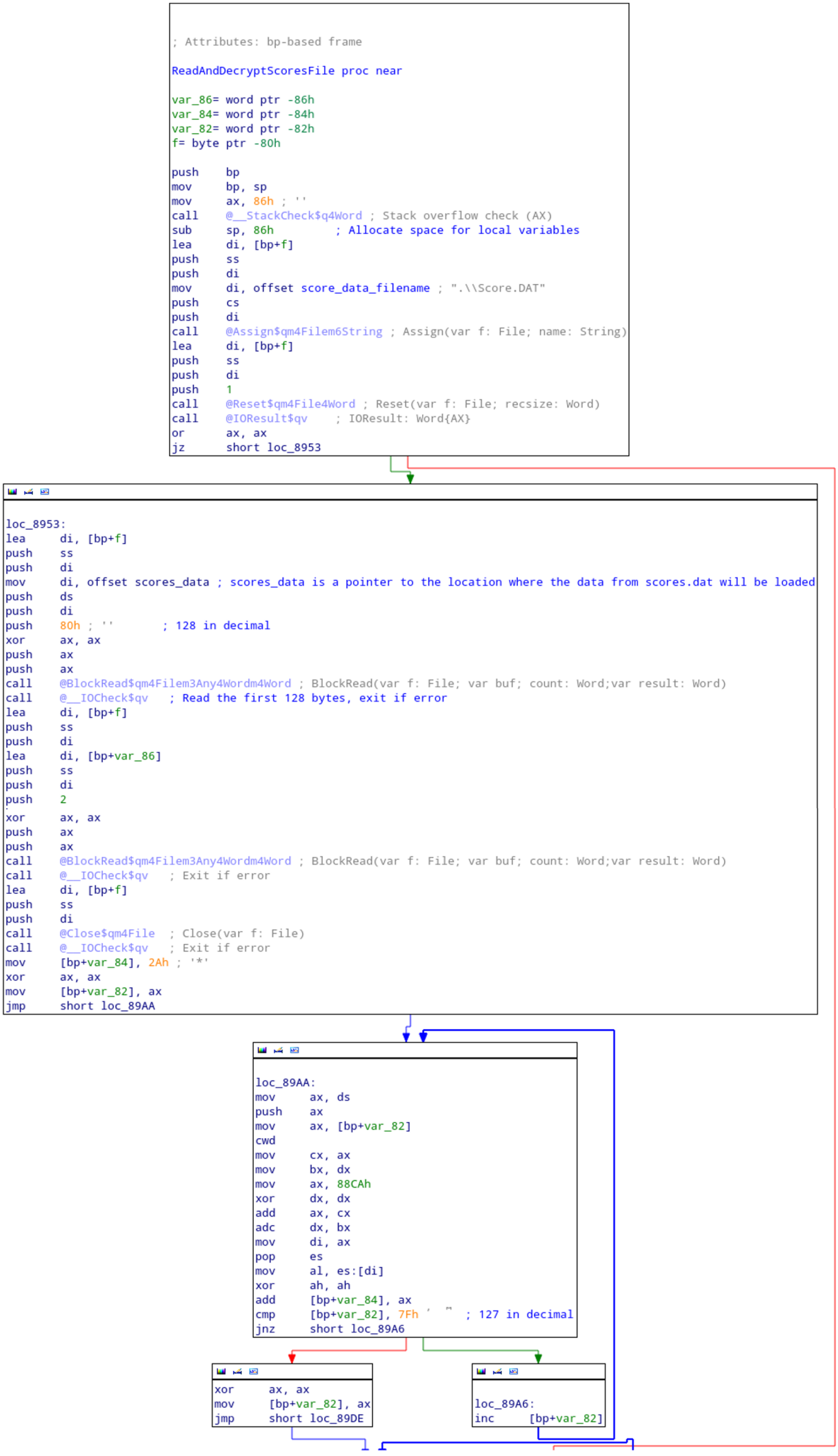
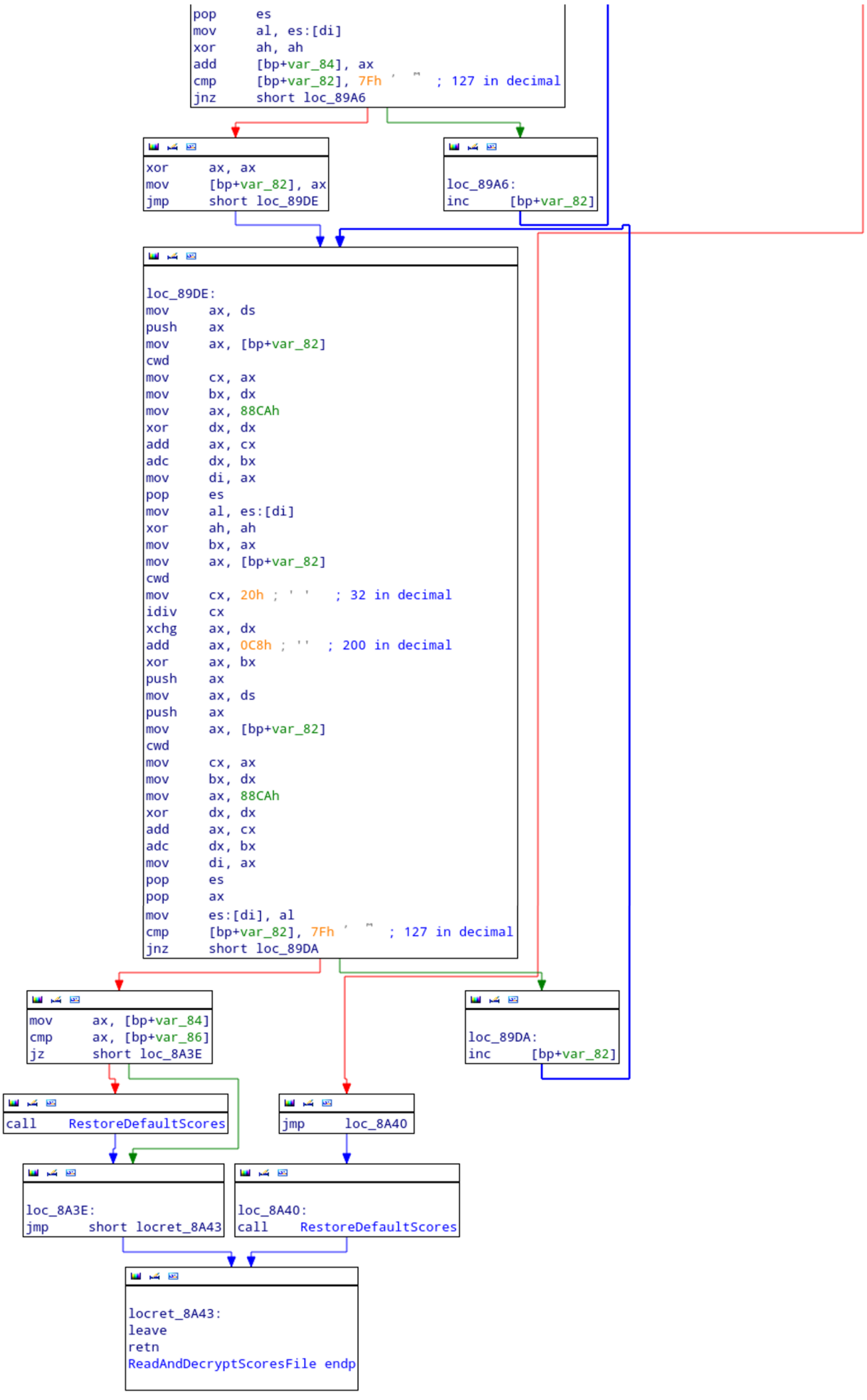

; Attributes: bp-based frame
ReadAndDecryptScoresFile proc near
var_86= word ptr -86h
var_84= word ptr -84h
var_82= word ptr -82h
f= byte ptr -80h
push bp
mov bp, sp
mov ax, 86h ; ''
call @__StackCheck$q4Word ; Stack overflow check (AX)
sub sp, 86h ; Allocate space for local variables
lea di, [bp+f]
push ss
push di
mov di, offset score_data_filename ; ".\\Score.DAT"
push cs
push di
call @Assign$qm4Filem6String ; Assign(var f: File; name: String)
lea di, [bp+f]
push ss
push di
push 1
call @Reset$qm4File4Word ; Reset(var f: File; recsize: Word)
call @IOResult$qv ; IOResult: Word{AX}
or ax, ax
jz short loc_8953
loc_8953:
lea di, [bp+f]
push ss
push di
mov di, offset scores_data ; scores_data is a pointer to the location where the data from scores.dat will be loaded
push ds
push di
push 80h ; '' ; 128 in decimal
xor ax, ax
push ax
push ax
call @BlockRead$qm4Filem3Any4Wordm4Word ; BlockRead(var f: File; var buf; count: Word;var result: Word)
call @__IOCheck$qv ; Read the first 128 bytes, exit if error
lea di, [bp+f]
push ss
push di
lea di, [bp+var_86]
push ss
push di
push 2
xor ax, ax
push ax
push ax
call @BlockRead$qm4Filem3Any4Wordm4Word ; BlockRead(var f: File; var buf; count: Word;var result: Word)
call @__IOCheck$qv ; Exit if error
lea di, [bp+f]
push ss
push di
call @Close$qm4File ; Close(var f: File)
call @__IOCheck$qv ; Exit if error
mov [bp+var_84], 2Ah ; '*'
xor ax, ax
mov [bp+var_82], ax
jmp short loc_89AA
loc_89AA:
mov ax, ds
push ax
mov ax, [bp+var_82]
cwd
mov cx, ax
mov bx, dx
mov ax, 88CAh
xor dx, dx
add ax, cx
adc dx, bx
mov di, ax
pop es
mov al, es:[di]
xor ah, ah
add [bp+var_84], ax
cmp [bp+var_82], 7Fh ; ''
jnz short loc_89A6
xor ax, ax
mov [bp+var_82], ax
jmp short loc_89DE
loc_89A6:
inc [bp+var_82]


pop es
mov al, es:[di]
xor ah, ah
add [bp+var_84], ax
cmp [bp+var_82], 7Fh ; 127 in decimal
jnz short loc_89A6
xor ax, ax
mov [bp+var_82], ax
jmp short loc_89DE
loc_89A6:
inc [bp+var_82]
loc_89DE:
mov ax, ds
push ax
mov ax, [bp+var_82]
cwd
mov cx, ax
mov bx, dx
mov ax, 88CAh
xor dx, dx
add ax, cx
adc dx, bx
mov di, ax
pop es
mov al, es:[di]
xor ah, ah
mov bx, ax
mov ax, [bp+var_82]
cwd
mov cx, 20h ; ' ' ; 32 in decimal
idiv cx
xchg ax, dx
add ax, 0C8h ; '' ; 200 in decimal
xor ax, bx
push ax
mov ax, ds
push ax
mov ax, [bp+var_82]
cwd
mov cx, ax
mov bx, dx
mov ax, 88CAh
xor dx, dx
add ax, cx
adc dx, bx
mov di, ax
pop es
pop ax
mov es:[di], al
cmp [bp+var_82], 7Fh ; 127 in decimal
jnz short loc_89DA
mov ax, [bp+var_84]
cmp ax, [bp+var_86]
jz short loc_8A3E
loc_89DA:
inc [bp+var_82]
call RestoreDefaultScores
jmp loc_8A40
loc_8A3E:
jmp short locret_8A43
loc_8A40:
call RestoreDefaultScores
locret_8A43:
leave
retn
ReadAndDecryptScoresFile endp