

\* rdi 는 고려 필요 x

\* rsp 는 push, pop. 연산을 통해 변화

rsp (앞 11자리 생략)

sub rsp, 128h (296 byte)

아래로 가라는 stack의 공간 확보

∴ 0xDF890 → 0xDF110

lea rbp, [rsp+20h] (32 byte)

bp를 sp + 20h로 잡기

∴ rbp = 0xDF190

(main bp)

input parameter에 3 입력

sub rsp, 0E8h (232 byte)

fast stack 내 공간 확보

∴ 0xDF150 → 0xDF668

lea rbp, [rsp+20h]

∴ rbp = 0xDF688

(fast bp)

0xDF8A0

0xDF898

0xDF890

0xDF190

0xDF110

0xDF168

0xDF160

0xDF158

0xDF150

0xDF688

0xDF668

push rbp → sp에 bp를 저장. (rbp=0)

push rdi → sp에 di를 저장 (rdi=0)



move dword ptr [input\_parameter], 3

복귀주소 = call 아래 mov 명령어 주소

push rbp

push rdi



① lea rsp, [rbp+0C8h (200 byte)]

rsp에 현재 rbp의 0C8h만큼 올려 (기존의 올라감)

②, ③ pop 연산 2회로 +8씩 두 번이 이루어지며

복귀주소로 복귀