При освобождаване на блок памет, той се слива в двойно свързан лист. Това се извършва от frontlink() macro(под Linux). Макрото съединява сегментите в намаляващ ред по големина.

В този случай недоброжелателят предоставя не адрес, а кратък код който цели да подлъже системата да изпълни функция предоставена от недоброжелателя вместо нейна.

Уязвим код:

#include <stdlib.h>

#include <string.h>

int main(int argc, char\*argc[]){

char \*first, \*second, \*third;

char \*fourth, \*fifth, \*sixth;

first = malloc(strlen(argv[2]) + 1)

second = malloc(1500);

third = malloc (12);

fourth = malloc (666);

fifth = malloc(1508);

sixth = malloc(12);

strcpy(first, argv[2]);// получава се препълване на буфера

free(fifth);// във forward pointera на петия блок се слага адресс към фалшив блок.

strcpy(fourth, argv[1]);

free(second);

return(0);

}

Във фалшивия блок е съхранен адрес към пойнтер на функция. Този пойнтер може да сочи към първата извикана деструктор функция(нейния адрес може да бъде намерен в сектора dtors на програмата). Недоброжелателят може да намери този адрес и да се опита да го замести с пойнтер сочещ към негова функция. Когато second се освободи frontlink() започва да го слепва към fifth блок. Резултата е че във forward пойнтера на fifth е записана адрес който сочи към функция и при извикването на return(0) вместо деструктор функция ще се извика друга предоставена от недоброжелателя.