**57. Управление на памет в конзолен режим и в Linux системи: структури в**

**паметта, повреждане на структурите при неправилно менажиране на памет.**

В повечето Linux системи за алокиране на паметта се използва принципът на Doug Lea,

който е доста бърз и ефикасен. Използва стратегия Best-Fit, т.е. пре-използва

освободените (free) парчета (chunks) памет със най-малки загуби. При освобождаване

се срастват парчета в по-големи.

Техниката на Doug Lea се базира на двустранно свързани списъци от

ОСВОБОДЕНИТЕ парчета.

При този метод, алокираните парчета имат следния формат: [size]{data}. [size] е

размерът на {data}, парчето. Примерно malloc(8) ще задели 8 байта + [size] и ще върне

адреса на парчето в паметта, където ще пише [8]{…}. Ако сме заделили повече парчета

памети ще изглеждат така: [8]{…} [32]{………} [4]{..} [128]{…………}, долепени едно

до друго (незадължително). Така ако си на първото парче и искаш да стигнеш до 3-то

парче ще скочиш един път 8 байта + още 32 байта надясно и ще стъпиш на 3-то парче.

Във [size] се включва и допълнителен бит - PREV\_INUSE накрая, който посочва дали

ПРЕДИШНОТО парче е освободено в момента.

Когато освободиме парче памет, ние само го маркираме че то е освободено. На мястото

на старта информация където е било{data} се записва друга служебна информация, а

именно forward указател и back указател, както и още веднъж [size]. По средата се

намира unused секция с неизползваема информация, останала от старите данни, може и

да липсва. Структурата на едно освободено парче памет приема следният вид:

free

Forward и back указателите сочат следващото/предишното свободно парче. Пази се

списък от освободените парчета, а не запазените. Когато се опита да се алокира ново

парче, започва да се обхожда списъкът докато не се намери подходящо свободно парче,

то се откъсва от списъка и върху което да алокира желаната памет. Използвайки това че

size 1

data

size 1

forward free

back

unused

size(същия)

са списък, допълнителния бит PREV\_INUSE и повтарящия се [size] накрая на

освободените парчета, може да се направи лесна дефрагментация на паметта, и две

съседни свободни парчета да бъдат слепени.

Пример: [8]{…} [32]{…free…} [8]{free} [4]{..} [8]{free} ще бъдат открити че стоят

заедно и ще бъдат сляти: [8]{…} [40]{…free…} [4]{..} [8]{free}.