1. Jakými typickými kroky prochází system pri otevření dosud neotevřeného souboru v rámci volaní open()? Předpokládáme, ze soubor je možné úspěšně otevřit (limity: cca 5 rozvitych vet) - 5 bodu Reseni Typicke kroky Ilustracni obrazek 1) Typicke kroky operace open() pri 1. otevření souboru 1. Vyhodnoti se cesta , posoudí existence a přistupova práva otevíraného souboru, nacte se jeho i-uzel z disku.

2. v tabulce v-uzlu se alokuje nova položka, do ni se vloži i-uzel a rozšíří se o některé dalsi udaje, vcetne poctu odkazu na danu v-uzel(nyní 1) Alokuje se nova polozka v tabulce otevřených souboru a naplní se odkazem na v - uzel a dalšími informacemi, zejmena: pozice v souboru, počet odkazu na zacatku 1), rezim otevření atd. Tabulka v-uzlu 4. Alokuje se nova (první volna) položka v tabulce deskriptoru, naplní se odkazem na v - uzel položku tabulky otevřených souboru 5. Vrati index alokovaneho deskriptoru (pripadne -1 pri chybe) Pocet odkazu Tabulka deskriptoruuu Tabulka otevřených souboru (ukazatel v paměti) stdin stdout stderr Odkaz ( hledárne první volnou položku. Nemusí vesk vártu mít říslo 3. nachř rento hude N

- 360.5 kB vyzaduj e 361 alokacních bloku ( protože se alokuje po blocích) ( alokaci blok - skupina pevného poctu sektoru, typicky 2<sup>2</sup>n, následujících řyzicky i logicky za sebou, která je nejmenší jednotkou diskového prostoru, kterou OS čte ci zapisuje pro běžných operacich

Předpokládejte velikost alokacniho bloku 1000B (1kB) a velikost odkazu na blok - 10B.
Kolik alokacnich bloku (bez i-uzlu) bude na disku s klasickým unixovym FS zabírat soubor o velikosti 360.5kB.
(Kolik prostoru to zabere (data + metadata) ?) Zďůvodně.

Reseni

366 = 361 data + 5 metadata

## ) - 10 datovych bloke bude odkazovano primymi odkazy - dalsich 1000/10 bloku je odkazovano pres nepřírný odkaz 1. úrovně z i-uzlu pres jeden - Gallsch 1000/10 dolku) je Odkazovano pres neprimy dolkaz 1. urovne z 1-uzu pres jeden pomocný adresovací blok - dale je použít nepřímý odkaz 2. úrovně v 1-uzů, vedoucí na jeden pomocny adresovací blok s nepřímý odkazy 1. úrovně. Takové jsou alokovany 3. Do nich se již podán uložít pozadovaných 251 zbyvajúcíh odkazu. - Celkove tedý bloku metadat (ky ruzove) i - uzlv D pomocny adresova S primymi odkazy 100 = 1000/10 neprimymi odkazy 1. PD)

>D

 Jaky je maximální počet čtení bloku z disku pro provedeni operaci
h = open( "/dir/symlink/", O\_RDONLY); read( h, buf, 10 ); read ( h, buf, 10 ); ? Predpokladejte přítom, ze adresáře jsou kratší než 1 blok, dir je adresar, symlink je symbolicky odkaz na soubor (dir/file, file je klasicky soubor delší než 20B zadny blok není na počátku ve vyrovnávací paměti, nedochazi k interferenci s dalšími procesy, prepnuti kontextu, prichodu signálu apod.

Obsah advesaru se vleze do iednoho bloku

## Reseni

7 čtení - při použití tzv. rychlého symlinku

- Na zacatku mame pouze číslo i-uzlu kořenového adresáře ze super-bloku, které lze převěst na odpovídající blok a ten bude načten získáme tak obsah i-uzlu kořenového adresáře \*f\* 1. čtení .
- 2. čtení načte obsah kořenového adresáře z bloku, na který je odkaz v i-uzlu kořenového adresáře "/" 2. čtení
- V datech "/" se nalezne dvojice ( dir, číslo i-uzlu dir ) a načte se blok s timto i-uzlem ( číslo bloku lze odvodit z čísla i-uzlu ) 3 čtení
- 4. čtení pak načte obsah adresáře "/dir" z bloku, na který je odkaz v j-uzlu "/dir" 4. čtení
- V datech "/dir" nalezneme číslo i-uzlu "/dir/symlink" a načteme blok s odpovidajícim i-uzlem 5. čtení
- Předpokladame-li použití rychlého symlinku, pak primo v i-uzlu "/dir/symlink" ie cesta "/dir/file"
- Preklad /dir na odpovídající i-uzel, tento i-uzel i jeho data mame ve vyrovnávací paměti není nutno znovu cest
- V datech "/dir/" se nalezne číslo i-uzlu "/dir/file a je načten blok s timto i-uzlem -6. čtení
- Pomoci odkazu z i-uzlu "/dir/file" je načten 1. blok tohoto souboru a z nej se vezme 10 bytu 7. čtení
- Další prikas read() již nečte z disku příslušna data isou ve vyrovnávací paměti



