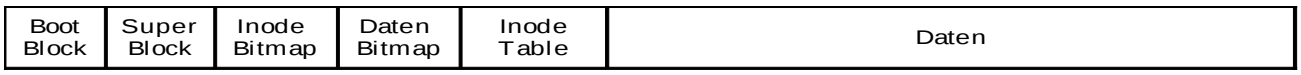
1. Unix-Dateisysteme

Unter Unix haben Dateisysteme grundsätzlich den folgenden logischen Aufbau:



- Bootblock: Startroutine des Betriebssystems
- Superblock: Verwaltungsinformationen des ganzen Dateisystems
- Datenblock-Bitmap: Für jeden Datenblock ein Bit (frei/nicht frei)
- Inode-Bitmap: Für jeden Inode ein Bit: benutzt/nicht benutzt
- Inode-Tabelle: Hier liegen alle Inodes mit den Verwaltungsinformationen zu Dateien
- Datenblöcke: die eigentlichen Nutzdaten, also Dateien und Verzeichnisse

Welche Datenblöcke von einer Datei belegt sind, wird in der Datenstruktur „Inode“ gespeichert, die es zu jeder Datei gibt. Sie bietet aber nur Platz für wenige direkte Verweise auf Datenblöcke – darum arbeiten Unix-Dateisysteme mit indirekter Adressierung: Ein einfach indirekter Eintrag zeigt auf einen Block, der weitere Adressen von Datenblöcken enthält; ein zweifach indirekter Eintrag zeigt auf einen Block, der einfach indirekte Einträge enthält usw.:

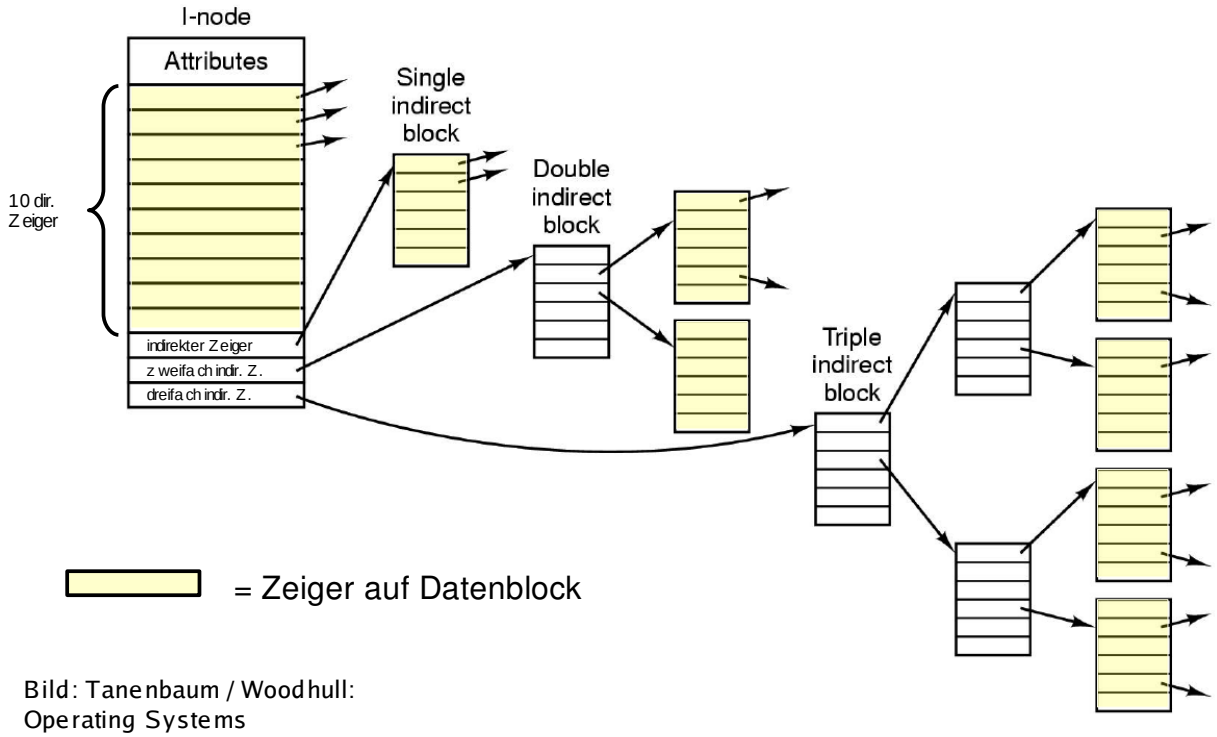


Bild: Tanenbaum / Woodhull: Operating Systems

Die Indirektion ermöglicht es, sehr große Dateien (aus vielen Blöcken bestehend) anzulegen.

Um die maximale Dateigröße in einem Dateisystem zu bestimmen, benötigen Sie folgende Informationen:

- Blockgröße (z. B. 32 KByte)
- Dateisystemgröße (z. B. 16 GByte)
- Anzahl der direkten, 1-fach indirekten, 2-fach indirekten etc. Verweise in einem Inode (z. B. 9 direkte, 4 einfach indir., 2 zweifach indir. und 1 dreifach indir.)

Es ergibt sich dann folgendes Rechenschema:

- Dateisystem: 16 GByte, Block: 32 KByte ⇒ Es gibt 16 G / 32 K = 0,5 M = 512 K = 2^{19} Blöcke. Blockadressen sind also mind. 19 Bit lang, in der Praxis: 4 Byte (kleinste 2er-Potenz an Bits, in die 19 Bit passen)
- In einen Block passen N := (Blockgröße / Adresslänge) = 32 KByte / 4 Byte = 8 K = 8192 = 2^{13} Adressen.
- Die maximale Dateigröße berechnen Sie als Produkt aus der Anzahl der adressierbaren Blöcke mit der Blockgröße. Die Anzahl berechnen Sie im Beispiel wie folgt:

$9 \times N^0$	9×1	(direkte)
$4 \times N^1$	4×8192	(1-fach indirekte)
$2 \times N^2$	$2 \times 8192 \times 8192$	(2-fach indirekte)
$1 \times N^3$	$1 \times 8192 \times 8192 \times 8192$	(3-fach indirekte)
	549.890.064.393	(Summe)

Dieser Wert, multipliziert mit der Blockgröße (32 KByte) ergibt dann 17.596.482.060.576 KB (ca. 16388 Terabyte).

[☛ Lösungsblatt]

a) Schlagen Sie den (deutschen oder englischen) Wikipedia-Eintrag zu „Inode“ nach und geben Sie eine kurze Erklärung, worum es sich bei Inodes handelt.

b) Erklären Sie, was es mit dem „Link Count“ in einem Inode auf sich hat und wie sich unter Unix Soft Links (symbolische Links) und Hard Links unterscheiden. (Auch die Antwort auf diese Frage müssen Sie online recherchieren.)

c) Rechenaufgabe: Ein Unix-Dateisystem habe folgende Eigenschaften:
– Blockgröße: 16 KByte
– Dateisystemgröße: 1 GByte
– 2-fache Indirektion; 10 direkte Verweise, 3 einfach indir. und 1 zweifach indir. Verweis
Berechnen Sie (wie oben) die maximale Dateigröße.

d) Ein Dateisystem arbeite mit dreifach indirekter Adressierung (wie im Beispiel). Warum führt eine Verdopplung der Blockgröße zur Ver-16-fachung der maximalen Dateigröße (also Faktor 2^4)?

e) Leiten Sie eine allgemeine Formel her – wenn ein Dateisystem mit n-facher Indirektion arbeitet und Sie die Blockgröße um den Faktor 2^k verändern[1], wie ändert sich dann die maximale Dateigröße?

[1] k kann negativ oder positiv sein; negative k stehen für Verkleinerungen, z. B.: k=-2 entspricht dem Reduzieren der Blockgröße auf ein Viertel (2^{-2} = ¼) des ursprünglichen Werts.

Lösungsblatt – Übung 8

Punkte:

	Name	**Matrikelnummer**
Teilnehmer 1		
Teilnehmer 2		

1a)

1b)

1c)

1d)

1e)