Linux Dateisysteme

Inhalt **TREETOOL IM HINTERGRUND LADEN!!!**

- Aufgaben & Funktionen von Dateisystemen
- Grundlagen
- Beschreibung von
  - + ext2 und ext3
  - + jfs
  - + xfs
  - + swap
- Vergleich: ext3 – jfs - xfs

Linux Dateisysteme

Inhalt

- Aufgaben & Funktionen von Dateisystemen
- Grundlagen
- Beschreibung von
  + ext2 und ext3
  + xfs
  + swap
- Vergleich: ext2 - ext3 – jfs – reiserFS - xfs

Aufgaben

- Dateinamen und Attribute (Flags & Dates)
- Strukturierung der Dateien
- Zugriffsschutz, Rechte
- gleichzeitiger Zugriff auf Dateien (lock-Flag)
- Dateitypen
  - + Datei
  - + Verzeichnis
  - + E/A-Gerät (Drucker, Terminal, Network)
- Links
- physikalische Speicherung von Daten

Funktionen

- erstellen / löschen
- kopieren / verschieben / umbennen
- öffnen / schließen
- lesen / schreiben
- anhängen
- setzen / auslesen von Attributen

Grundlagen

Festplatte

- Speicherung auf übereinanderliegenden Magnetscheiben
- diese sind in mehrere tausend Spuren unterteilt
- alle gleichen Spuren bilden einen Zylinder
- Spuren sind in Sektoren (i.d.R. 512 Byte + Prüfsumme) eingeteilt
- Dateisystem faßt mehrere Sektoren zu einem Block, was dann die kleinste Einheit für den User ist, zusammen (bei MS Windows Cluster genannt)

Grundlagen

Master Boot Record

- erster Sektor der Festplatte
- beinhaltet ausführbaren Code
- Bootloader
- Partitionstabelle
  + beginnt ab Byte 446
  + 64 Byte groß
  + 4 Einträge mit je 16 Byte, daher 4 primäre Partitionen
- kopieren des MBR ins aktuelle Verzeichnis

```
dd if=/dev/hda of=mbr.img count=1 bs=512
```

- und dann anzuschauen mit

```
od -x mbr.img
```

Grundlagen

Partitionen

- 4 primäre Partitionen
- Partition kann vom Typ “erweitert” sein
  --> beherbegen logische Laufwerke
- Ausschnitt aus meinem MBR

```
0000700 0001 fe83 023f 003f 0000 bc04 0000 0000
0000720 4d01 fe83 ffff e00d 0012 acce 0381 0000
0000740 0301 fe83 4c3f bc43 0000 23ca 0012 0000
```

- Ausgabe von fdisk -l /dev/hda

```
   Device Boot      Start         End      Blocks   Id  System
/dev/hda1               1           3       24066   83  Linux
/dev/hda2              78        3739    29415015   83  Linux
/dev/hda3               4          77      594405   83  Linux
Partition table entries are not in disk order
```

Grundlagen

i-nodes

- i-nodes stehen ziemlich am Anfang des Dateisystems
- jedes Objekt im Dateisystem hat einen Indexknoten
- dieser i-node zeigt auf den Block, in dem die Daten zu dem zugehörigen Objekt gespeichert sind (auch seine Metadaten, außer dem Namen)
- i-nodes besitzen Verweise auf direkte und indirekte Blöcke

ext2

Superblock

beschreibt die Konfiguration des Dateisystems

- primärer Superblock ist ab Byte 1024 Byte zu finden
- mehrere Sicherheitskopien auf dem Dateisystem verteilt

Informationen:

- Bitmap über Belegung von Blöcken und i-nodes
- wann wurde das Dateisystem
  + angelegt (von welchem OS)
  + verändert
  + ein- bzw. Ausgehängt
- Anzahl Blöcke und i-nodes im Dateisystem, sowie in den Blockgruppen

ext2

Blockgruppen

- mehrere Blöcke bilden eine Blockgruppe
- jede Blockgruppe enthält:
  + Kopie des Superblocks
  + Kopie aller Blockgruppendeskriptoren
  + Blockbelegungsbitmap für diese Blockgruppe
  + Inodebelegungsbitmap für Inodes dieser Blockgruppe
- die zwei Bitmaps definieren die Blockgruppengröße, da sie je einen Block füllen
- bei Blockgröße 1024 Bytes kann eine Blockgruppe maximal 8192 Inodes und 8192 Blöcke enthalten
- bei Blockgröße 1 kB ist Blockgruppe demnach 8 MB groß

ext2

Deskriptoren

- ein Deskriptor ist 32 Byte groß
- er beschreibt eine Blockgruppe:
  + Adressen der Blockbelegungsbitmap und der Inodebelegungsbitmap
  + Startadresse der Inodetabelle für seine Blockgruppe
  + statistische Informationen über die Auslastung (freie Blöcke, freie i-nodes, Verzeichniszahl)
- Deskriptortabelle mit allen Deskriptoren liegt hinter dem Superblock

Bitmaps

- hinter der Deskriptorentabelle liegt die Block- und dann die i-node Bitmap

ext2

Anlegen von Dateien

- i-node einer Datei soll in selber Blockgruppe, wie die i-node vom dazugehörigen Verzeichnis "landen"
- Daten sollen nach Möglichkeit auch in dieser Blockgruppe liegen
- neue Verzeichnis kommen in eine neue Blockgruppe, in die mit überdurschnittlich vielen freien i-nodes und davon in die mit den meisten freien Datenblöcken
- es verwendet hierzu einen Belegungsalgorithmus, der mit einem Zielblock arbeitet
- ist dies nicht möglich, wird in der Nähe des Ziels gesucht
- scheiter auch dies, beginnt eine lineare Suche

ext2

Verringerung von Fragmentierungen

- ext2 versucht Daten in möglichst großen, zusammenhängenden Stücken (extends) zu schreiben
- geöffnete Dateien versuchen 8 zusammenhängende Blöcke für sich zu reservieren, bis sie geschlossen werden

Auslesen von Daten

- ext2 liest mehr ein als der User will um den Cache zu füllen
- viele Leseanforderungen an den Gerätetreiber werden zu größeren Leseanforderungen zusammengefasst

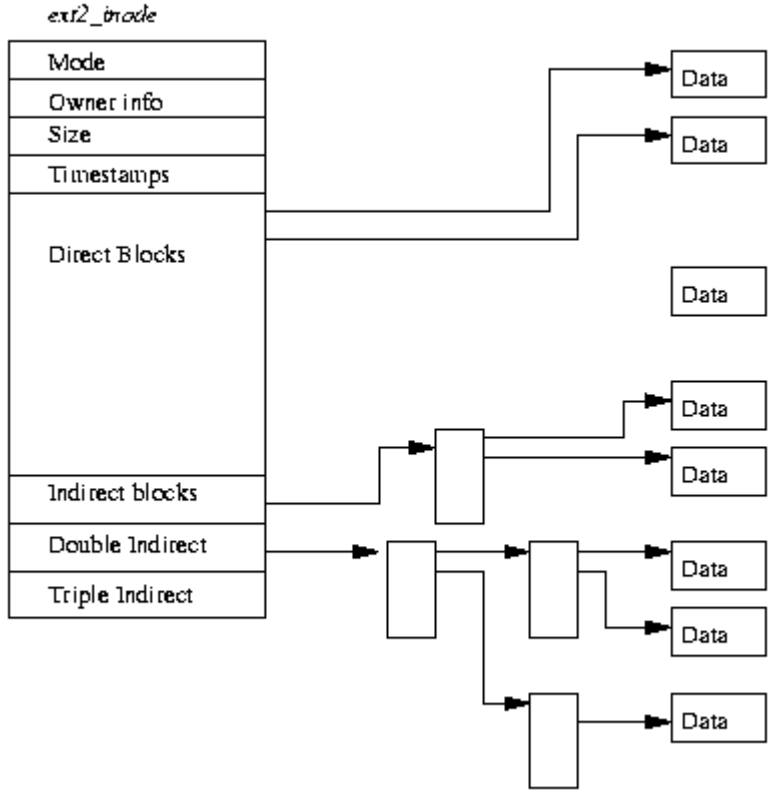
ext2

i-node Tabelle

- liegt hinter den Bitmaps, danach kommen die Daten
- beinhaltet alle i-nodes

i-node

- *mode* beschreibt Zugriffsrechte sowie Art des Objekts
  - + Datei
  - + Verzeichnis
  - + Link
  - + Blockgerät
  - + Zeichengerät
  - + FIFO

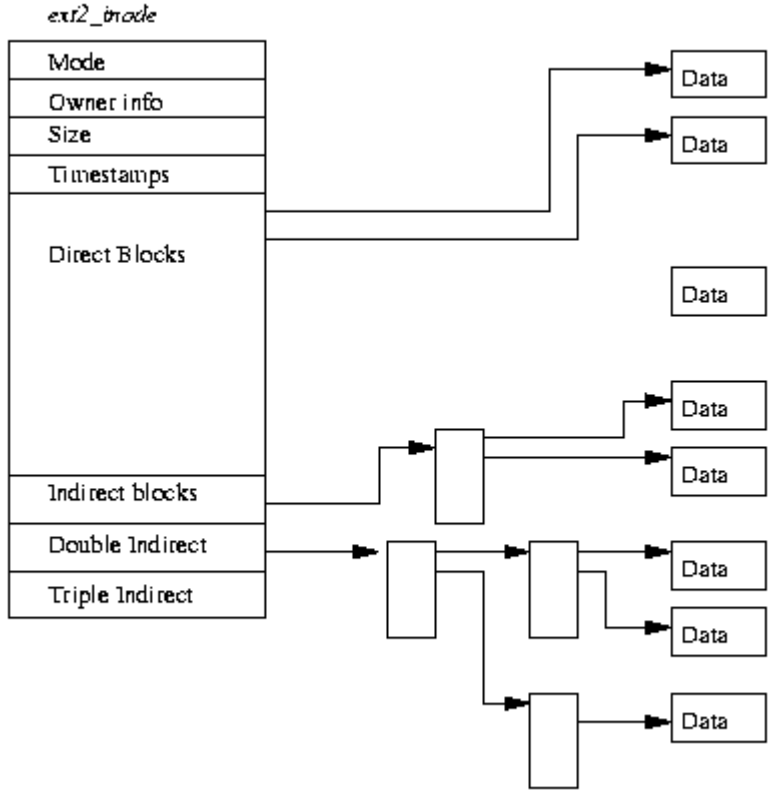


ext2

i-node

- *Owner info* UID des Besitzers
- *Size* Größe in Bytes
- *Timestamps* Änderung und Erstellung
- addressiert 12 direkte Blöcke
- und zusätzlich je einen
  - indirekten
  - zweifach indirekten
  - und dreifach indirekten

  Block



ext2

Größen

Blockgröße:	*1 kB*	*2 kB*	*4 kB*	*8 kB*
max. Dateigröße:	16 GB	256 GB	2048 GB	2048 GB
max. Dateisystemgröße:	2047 GB	8192 GB	16384 GB	32768 GB

ext2

Verzeichnisse

- spezielle Dateien
- enthalten anstatt Link auf Daten im ersten direkten Block einen Link auf sich selber .
- im zweiten direkten Block steht der Link zum Übergeordneten Verzeichnis ..

ext3

Journaling File System

- erweitert ext2 um ein Journal
- ist abwärtskompatibel zu ext 2
- das Journal ist eine reguläre Datei
- Änderungen am Dateisystem z.B. Namensänderung werden gelogt
- stürzt während der Änderung der PC ab, so kann mit Hilfe der Logdatei die Änderung verworfen werden und die ursprüngliche Datei ist wieder vorhanden
- Journal garantiert nicht, dass alle Änderungen geschrieben sind. Kernelpuffer werden nicht berücksichtigt

--> garantiert nur vernünftige Daten, nicht aktuelle Daten

Grundlagen

Journal

- logt Veränderungen im Dateisystem z.T in Dateien
- Metadaten- (Dateisystem gesichert) Full-Journaling (alles)
- alte Dateien bleiben solange gültig, bis der Schreibvorgang abgeschlossen ist
- bei Systemabsturz muß die Festplatte auf fehlerhafte Daten geprüft werden
- hat das Dateisystem ein Journal, so ist bekannt welche Dateien bzw. Partitionen verändert wurden

--> es muß nicht alles überprüft werden

--> Zeitersparnis

XFS

Verwaltet:

- bis zu 9 Exabyte (9 Mio Terabyte) große Dateien

- 18 Exabyte große Partitionen

- 512 Byte bis 1MB (für Echtzeit) große Datenblöcke

- "Blockgruppen" können bis zu 4 GB groß sein

XFS

64-Bit Dateisystem

- alle globalen Counter
- iNode – Nummern
- Blocknummern

sind 64 Bit lang.

Journaling

Größe der Partition kann im gemounteten Zustand verändert werden

Organisation der Blöcke, Bitmaps und Verzeichnisse in b-Bäumen

Grundlagen

b-Bäume

- werden in der Informatik als Indexstruktur zum schnellen Zugriff auf Daten verwendet

treetool

kleine Dateien

- werden direkt in den iNodes gespeichert
- verschwendet kein Speicherplatz bei großen Blockgrößen
- schneller Zugriff

Bandbreitenverwaltung

- Programme können sich Bandbreite auf das FS reservieren
- ideal für video on demand, Satellitensender....

jfs und reiserFS

- zwei sehr ähnliche Dateisysteme
- führen ein Journal
- arbeiten mit Bäumen

Unterschied zu XFS

- keine 64 Bit Adressierung
- keine Bandbreitenverwaltung

swap

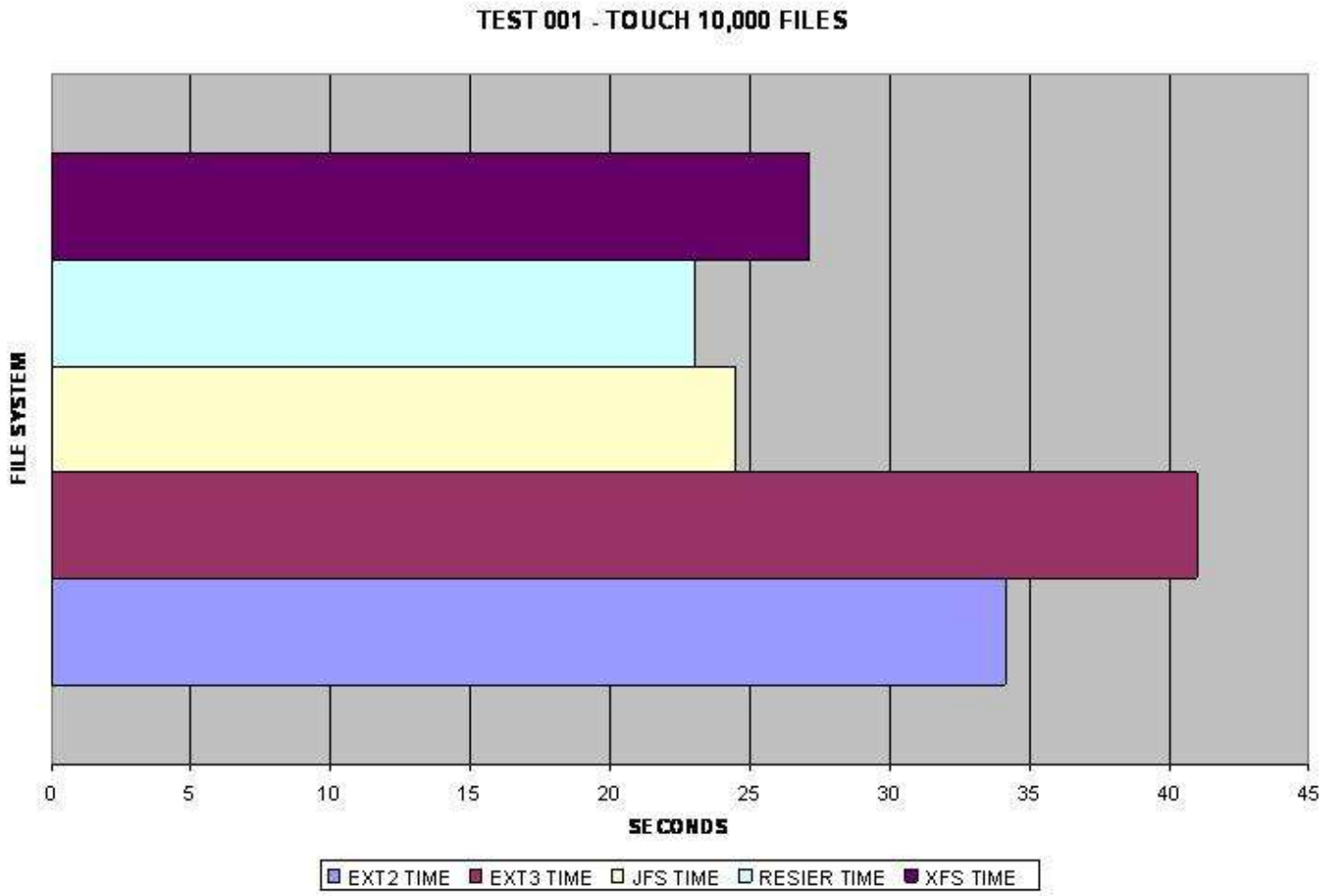
- Wenn der Arbeitsspeicher voll ist, wird auf die swap Partition ausgelagert

- unter Windows wird in eine Datei ausgelagert

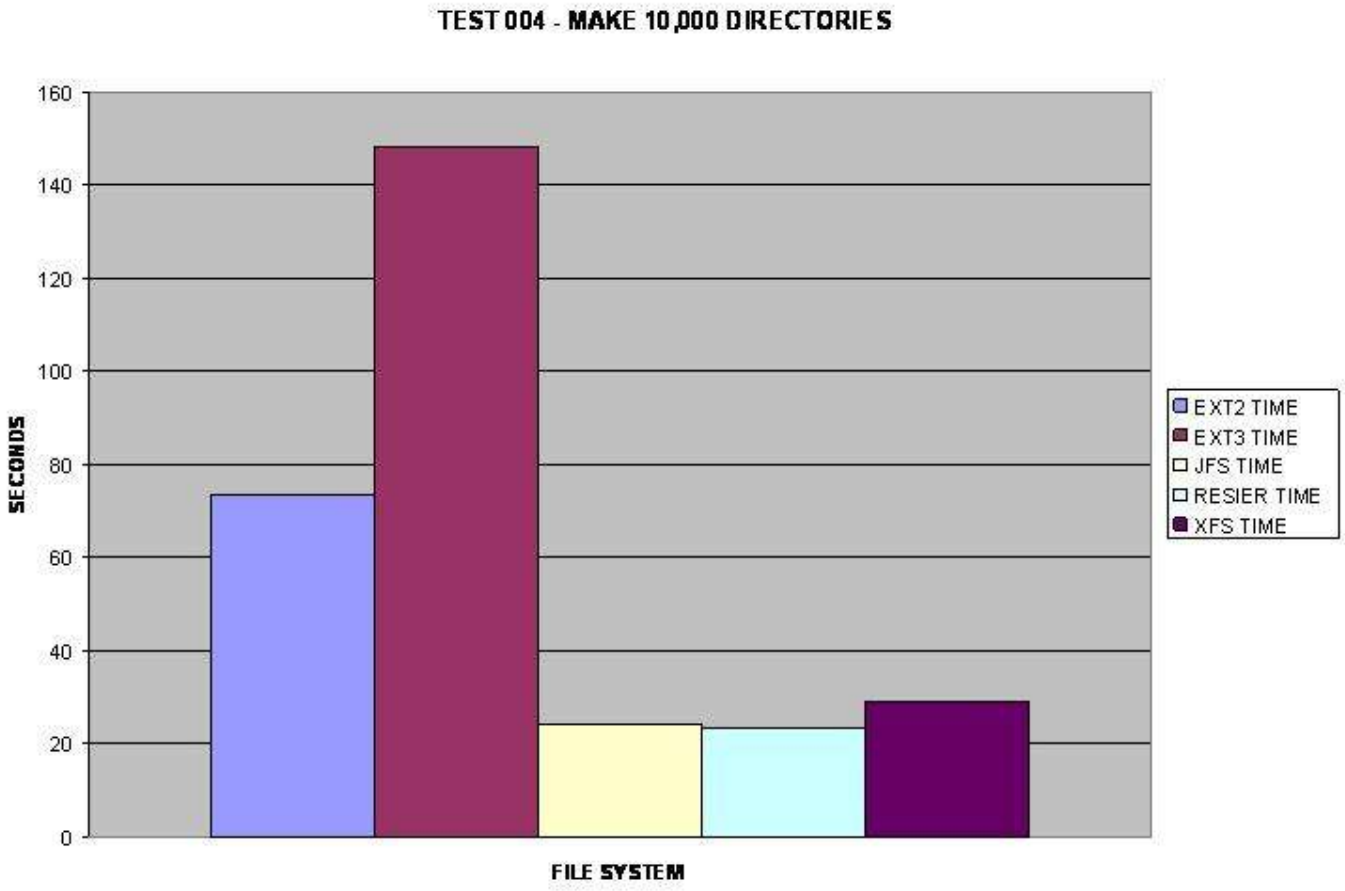
- unter Linux in der Regel in eine eigene Partition

- swap wird nicht gemountet, sondern
  mit swapon <partition> eingeschalten
  und mit swapoff <partition> wieder deaktviert

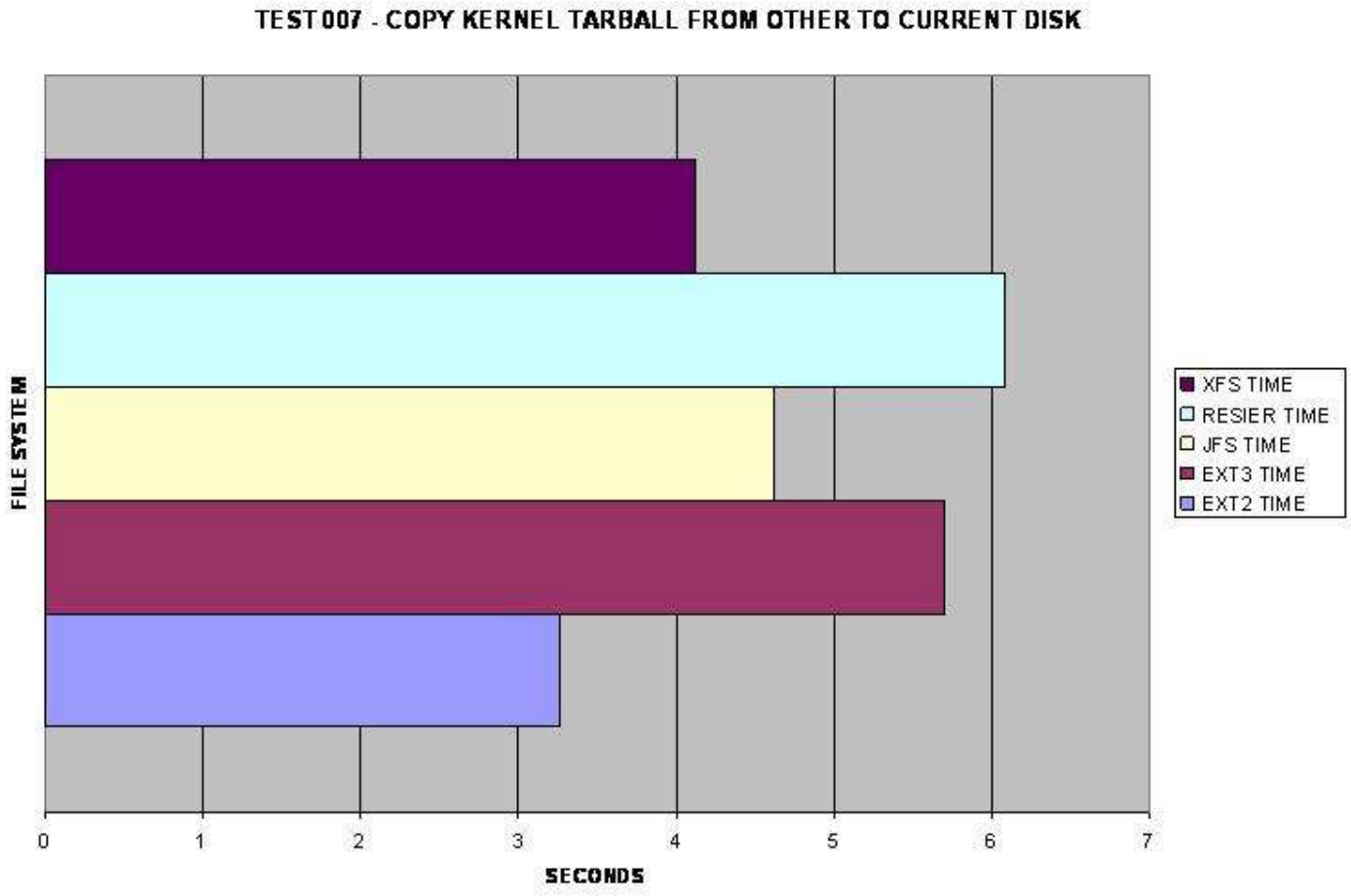
Vergleich - Zugriffszeiten



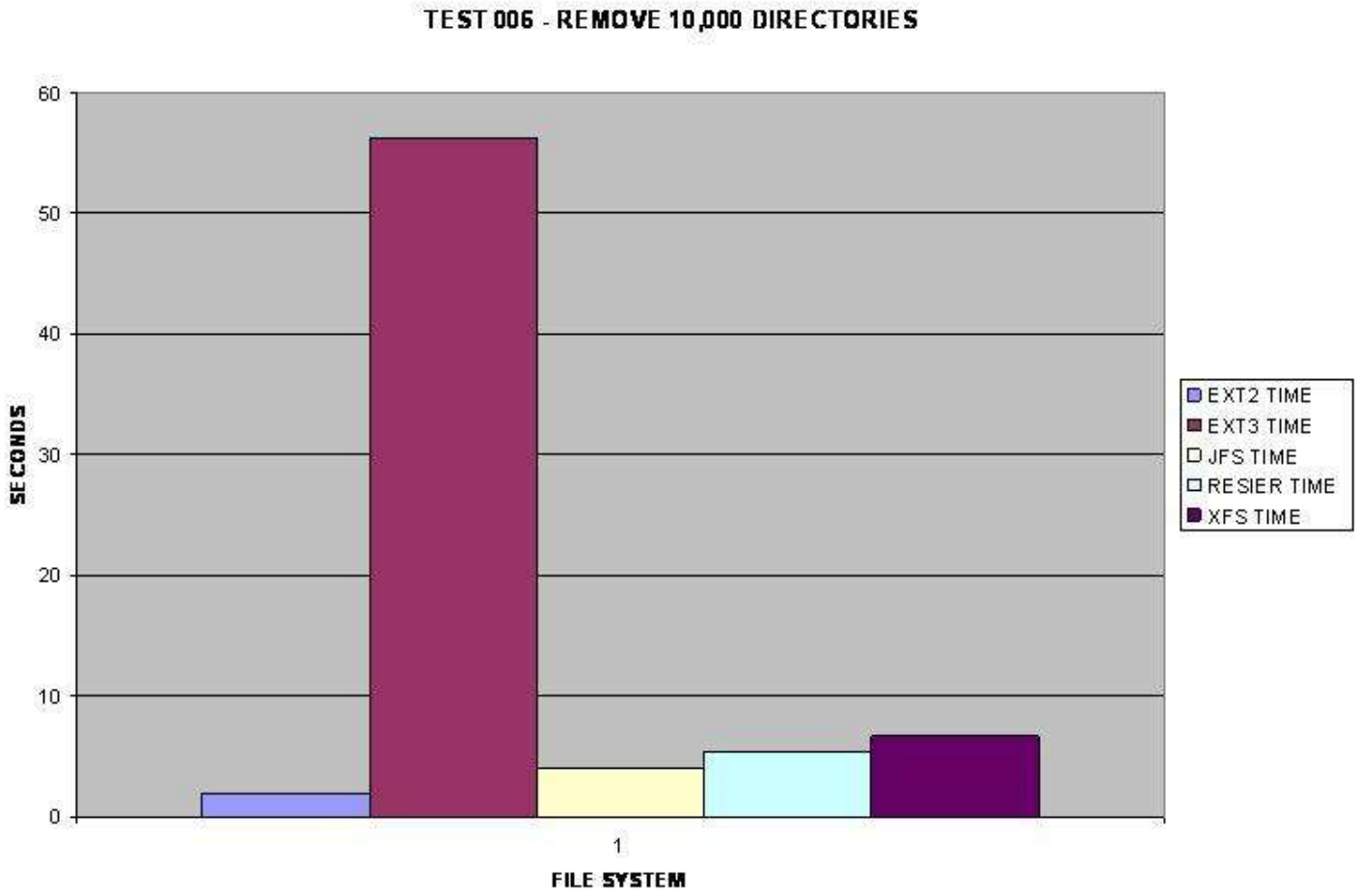
Ordner erstellen



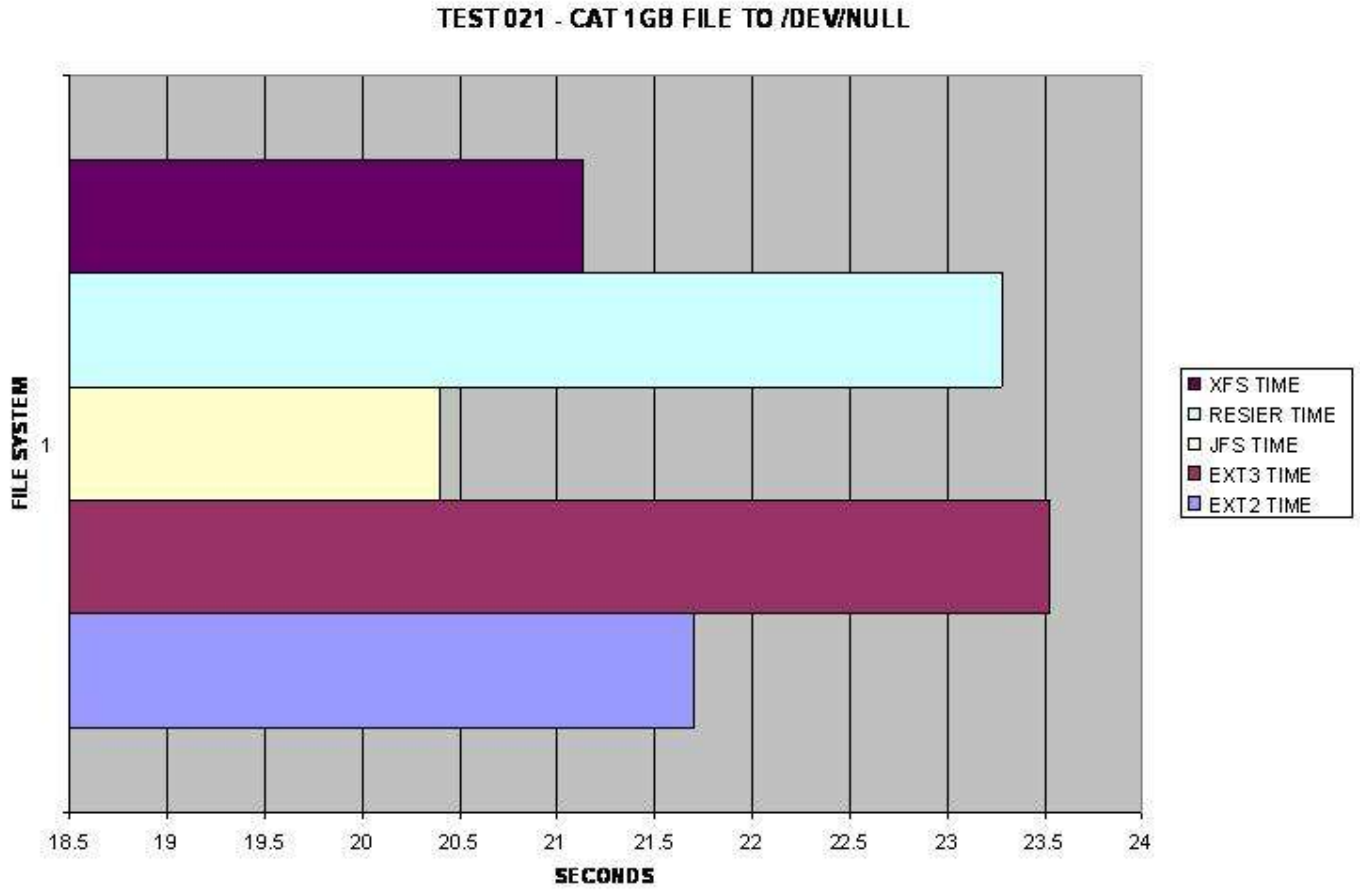
eine größere Datei schreiben



Ordner löschen



eine größere Datei auslesen



Gesamtvergleich