LTFS en Tape

Hardware

Als eerste kun je controleren of de PCI-kaart geïnstalleerd is. Dat doe je met het volgende commando:

```
lspci
```

Daar kun je iets zien van bijv. Fibre Channel …

Met het volgende commando kun je zien welke devices er zijn. Dit kunnen bijvoorbeeld harddisks, maar ook de tapestreamers zijn:

```
lsscsi -g
```

Je ziet hier in de laatste twee kolommen zowel een stx en sgx. Dit zijn twee interfaces naar dezelfde hardware.

1. STx = tape-interfacae : data/backups: (st is niet terugspoelen, nst wel, st0 = nst0 / st1 = nst1 / etc).
2. SGx = direct SCSI : Status/beheer: vaak sg. Denk aan logs opvragen etc. dus ook status van drive etc.

Je kunt sgY niet simpelweg “omrekenen” naar stX op basis van nummers; die tellers lopen onafhankelijk. De koppeling maak je via hetzelfde SCSI-adres (zoals ik eerder liet zien met sysfs/udev).

Tapedrive

Om informatie over de drive zelf uit te lezen kun je het volgende doen:

```
sudo sg_logs -a /dev/sg3
```

Dan krijg je mogelijk de volgende output:

```
Device statistics log page (ssc-3 and adc)
  Lifetime media loads: 2200
  Lifetime cleaning operations: 6
  Lifetime power on hours: 61183
  Lifetime media motion (head) hours: 11204
  Lifetime metres of tape processed: 132126786
```

Wanneer het lang geleden is dat een tape gereinigd is, kun je een cleaning-tape laden. Dan wordt de drive automatisch gereinigd.

Als je benieuwd bent hoe oud een tape is, dan kun je dat uitlezen met:

```
./tape-age.sh
```

Vrije ruimte kun je zien met:

```
df -h /mnt/ltfs
```

Er is een script waarmee je een test kunt doen om data te schrijven naar en lezen van tape. Hiervoor moet je een tape in de drive doen en het volgende commando geven (voor een test met 50Gb).:

```
sudo TAPE_DEV=/dev/nst0 SCSI_DEV=/dev/sg3 TEST_SIZE_MIB=51200 ./testdrive.sh
```

De map /mnt/data1/tmp/tapedrive moet bestaan en er moet minstens 100Gb geschreven kunnen worden.

WORM

Op een WORM-tape kun je geen LTFS schrijven. Daarom schrijven met bijv. tar. Hiervoor moet je eerst naar het einde van de tape omdat je vanaf daar met het tar-commando kunt schrijven:

```
sudo mt -f /dev/nst0 eod
```

Vrije ruimte

Vervolgens kun je uit de sg_logs informatie halen welke grootte nog vrij is. Daarvan kun je het beste wat extra ruimte vrijhouden.
Er is een sh-bestandje (onderaan de bron) waarmee de grootte getoond wordt. Dit gaat met het volgende commando:

```
./list_size.sh
```

Doorlopen van de tape

Terugspoelen naar begin van de tape

```
mt -f /dev/nst0 rewind
```

Tonen welke tar op die plek staat

```
tar -tvf /dev/nst0
```

Naar volgende en vorige set

```
mt -f /dev/nst0 fsf 1
```

```
mt -f /dev/nst0 bsfm 1
```

Daarna weer tonen.

Naar einde

Met dit commando kun je naar het einde van de tape schrijven:

```
mt -f /dev/nst0 eod
```

Data schrijven

Daadwerkelijk schrijven van de data doe je met het tar commando. Dat gaat als volgt:

```
sudo tar -cvf /dev/nst0 "/mnt/data/sessions.zip"
```

of meerdere bestanden als één set:

```
sudo tar -cvf /dev/nst0 "/mnt/data/sessions.zip"
"/mnt/data/sessions_bestandslijst.txt"
```

Anders worden het losse sets.

Tape afsluiten

Je kunt aan het einde van de tape een EOF schrijven:

```
sudo mt -f /dev/nst0 weof 1
```

Tape terugspoelen

Terugspoelen kan handmatig door het volgende commando

```
sudo mt -f /dev/nst0 rewind
```

Data teruglezen

De data teruglezen vanaf tape gaat als volgt:

```
sudo mt -f /dev/st0 rewind
sudo tar -xvf /dev/st0 -C /mnt/restore "mnt/data/music_bestandslijst.txt"
```

LTFS

Een tape als LTFS gebruiken kan. Dan moet je wel een aantal dingen doen. Hiervoor heb je de volgende commando's nodig: ltfs mkltfs ltfsck

Dit kun je controleren met:

```
which ltfs mkltfs ltfsck
```

LTFS Tape formatteren

De tape wordt geformatteerd met het volgende commando:

```
sudo mkltfs -d /dev/sg3
```

Zet er -f achter als je zeker weet dat je alles weg wilt gooien.

Vervolgens kun je een volumenaam aangeven:

```
sudo mkltfs -d /dev/sg3 -n "KN0XX"
```

Tape mounten

Vervolgens kun je de tape mounten met het volgende commando:

```
ltfs -o devname=/dev/sg3 /mnt/ltfs
```

Tape unmounten

Aan het einde moet je altijd de tape unmounten. Dat gaat met het volgende commando:

```
sudo umount /mnt/ltfs
```

Daarna moet je de tape offline halen:

```
sudo mt -f /dev/nst0 offline
```

Dan kun je controleren of de tape 'offline' is. Dat gaat via:

```
sudo mt -f /dev/nst0 status
```

Handige kopieeropdrachten enzo

Als je wilt zien hoe het kopiëren gaat dan kun je pv gebruiken. Eerst installeren:

```
sudo dnf install pv
```

Daarna kun je het volgende commando gebruiken om iets te kopiëren:

```
pv -s "$(stat -c%s /mnt/ltfs/homes.zip)" /mnt/ltfs/homes.zip >
/mnt/data/homes.zip
```

Daarbij wordt de tool gestart met de grootte waardoor je kunt zien hoe lang het gaat duren.

Dit commando heb ik in een copy2.sh gezet waarmee ik niet alle informatie hoef te kennen. De broncode staat beneden.

Je kunt met één commando ook van alle subfolders in een map een losse zip-file maken. Dat gaat als volgt:

```
mkdir -p zips && for d in */; do zip -rv "zips/${d%/}.zip" "$d"; done
mkdir -p zips && for d in */; do [[ "$d" != "zips/" ]] && zip -rv
"zips/${d%/}.zip" "$d"; done
```

Zien hoeveel ruimte vrij is op een bepaalde mount gaat als volgt:

```
df -h /mnt/data
```

Typische kopieeropdracht voor archiveren

Typisch betekent dit het volgende:

1. ZIP-Files maken van alle shares op de NAS
2. Tape laden met Robot
3. Dan mounten : ltfs -o devname=/dev/sg3 /mnt/ltfs
4. Dan map aanmaken waar bestanden naartoe moeten : mkdir /mnt/ltfs/202xx (2026b voor tweede in 2026 bijv.)
5. Dan kopieren: /home/gmk/copy2.sh (met TAB selecteren van meerdere files)
6. Tussendoor controleren of de tape niet verder dan 80-85% vol zit met : df

Tapeloader

Je kunt de tapeloader uitlezen met het volgende commando :

```
sudo mtx -f /dev/sch0 status
```

Broncodes

list_size.sh

```
sudo sg_logs -p 0x31 /dev/sg3 \
| awk -F': ' '
/Main partition remaining capacity/ {rem=$2}
END{
  safe=rem*0.90
  printf "Vrij:   %.1f GiB (%.1f GB)\n", rem/1024, rem/1024*1.073741824
  printf "Veilig: %.1f GiB (%.1f GB)  [10%% marge]\n", safe/1024,
safe/1024*1.073741824
}'
```

list_all_sets.sh

```
#!/usr/bin/env bash
set -euo pipefail

DEV=/dev/nst0

sudo mt -f "$DEV" rewind

i=1
while true; do
  echo
  echo "===== SET $i ====="
  if ! sudo tar -tvf "$DEV"; then
    echo "Einde tape of geen tar-set meer."
    break
  fi
  # spring naar volgende set (filemark overslaan)
  if ! sudo mt -f "$DEV" fsf 1; then
    echo "Kan niet verder spoelen (waarschijnlijk EOT)."
    break
  fi
  i=$((i+1))
done
```

testdrive.sh

```
#!/usr/bin/env bash
set -euo pipefail

############################
# LTO-6 drive/tape test
```

```
# Destructief: schrijft naar tape en verifieert readback.
#
# Vereisten:
#   dnf install mt-st sg3_utils coreutils
#
# Devices:
#   TAPE_DEV  : /dev/nst0  (non-rewind tape)
#   SCSI_DEV  : /dev/sg3   (sg generic)
############################

TAPE_DEV="${TAPE_DEV:-/dev/nst0}"
SCSI_DEV="${SCSI_DEV:-/dev/sg3}"

# Test parameters
TEST_DIR="${TEST_DIR:-/mnt/data/tmp/tapetest}"
TEST_FILE="${TEST_FILE:-$TEST_DIR/test.bin}"
READBACK_FILE="${READBACK_FILE:-$TEST_DIR/test_readback.bin}"
HASH_FILE="${HASH_FILE:-$TEST_DIR/test.bin.sha256}"

# Size in MiB (10240 MiB = 10 GiB)
TEST_SIZE_MIB="${TEST_SIZE_MIB:-10240}"
BS_MIB="${BS_MIB:-1}"

log() { echo -e "\n== $* =="; }

require_root() {
  if [[ $EUID -ne 0 ]]; then
    echo "Run dit script als root (sudo)."
    exit 1
  fi
}

check_cmds() {
  local missing=0
  for c in mt sg_logs dd sha256sum egrep awk; do
    if ! command -v "$c" >/dev/null 2>&1; then
      echo "Mist commando: $c"
      missing=1
    fi
  done
  [[ $missing -eq 0 ]] || exit 1
}

confirm_destructive() {
  cat <<EOF
WAARSCHUWING: Deze test schrijft ${TEST_SIZE_MIB} MiB naar ${TAPE_DEV}
en OVERSCHRIJFT ALLE DATA op de geladen tape.

Typ exact: YES  (zonder quotes) om door te gaan.
EOF
  read -r answer
```

```
  if [[ "$answer" != "YES" ]]; then
    echo "Afgebroken."
    exit 0
  fi
}

mt_status() {
  sudo mt -f "$TAPE_DEV" status || true
}

pre_checks() {
  log "Drive/tape status (mt)"
  mt_status

  log "TapeAlert (actieve flags)"
  if sudo sg_logs --page=0x2e "$SCSI_DEV" | egrep -i ": 1$" ; then
    echo "Let op: er staan TapeAlert flags aan."
  else
    echo "Geen actieve TapeAlerts."
  fi

  log "Read counters (voor test)"
  sudo sg_logs --page=0x03 "$SCSI_DEV" | egrep -i "Errors
corrected|rewrites|uncorrected|Hard read"

  log "Write counters (voor test)"
  sudo sg_logs --page=0x02 "$SCSI_DEV" | egrep -i "Errors
corrected|rewrites|uncorrected|Hard write"

  log "Device statistics (kerncounters)"
  sudo sg_logs --page=0x14 "$SCSI_DEV" | egrep -i "Lifetime cleaning|Hard
read errors|Hard write errors|since last successful cleaning|media
loads|power on hours" || true
}

generate_test_data() {
  log "Maak testdata (${TEST_SIZE_MIB} MiB) + checksum"
  mkdir -p "$TEST_DIR"
  rm -f "$TEST_FILE" "$READBACK_FILE" "$HASH_FILE"

  # gebruik /dev/urandom voor echte random testdata
  dd if=/dev/urandom of="$TEST_FILE" bs="${BS_MIB}M" count="$TEST_SIZE_MIB"
status=progress
  log "Genereren van hash"
  sha256sum "$TEST_FILE" | tee "$HASH_FILE"
}

write_to_tape() {
  log "Rewind + blocksize = variable"
  mt -f "$TAPE_DEV" rewind
  mt -f "$TAPE_DEV" setblk 0
```

```
  log "Schrijf testdata naar tape"
  dd if="$TEST_FILE" of="$TAPE_DEV" bs="${BS_MIB}M" status=progress

  log "Schrijf filemark en rewind"
  mt -f "$TAPE_DEV" weof 1
  mt -f "$TAPE_DEV" rewind
}

readback_and_verify() {
  log "Lees terug van tape"
  dd if="$TAPE_DEV" of="$READBACK_FILE" bs="${BS_MIB}M" status=progress

  log "Verifieer checksum"
  sha256sum -c "$HASH_FILE"
}

post_checks() {
  log "Read counters (na test)"
  sudo sg_logs --page=0x03 "$SCSI_DEV" | egrep -i "Errors
corrected|rewrites|uncorrected|Hard read"

  log "Write counters (na test)"
  sudo sg_logs --page=0x02 "$SCSI_DEV" | egrep -i "Errors
corrected|rewrites|uncorrected|Hard write"

  log "TapeAlert (actieve flags na test)"
  if sudo sg_logs --page=0x2e "$SCSI_DEV" | egrep -i ": 1$" ; then
    echo "Let op: er staan TapeAlert flags aan."
  else
    echo "Geen actieve TapeAlerts."
  fi

  log "Device statistics (na test)"
  sudo sg_logs --page=0x14 "$SCSI_DEV" | egrep -i "Hard read errors|Hard
write errors|since last successful cleaning|Lifetime metres|media
motion|activity duty cycle" || true
}

summary() {
  log "SAMENVATTING / INTERPRETATIE"
  echo "1) Checksum OK?  -> hierboven moet 'OK' staan."
  echo "2) Uncorrected read/write errors moeten 0 zijn."
  echo "3) TapeAlert na test liefst leeg."
  echo
  echo "Als je wilt dat ik oordeel geef: plak pre/post counters + TapeAlert
hier."
}

main() {
  require_root
  check_cmds
```

```
  confirm_destructive
  pre_checks
  generate_test_data
  write_to_tape
  readback_and_verify
  post_checks
  summary
}

main "$@"
```

tapeinfo.sh

```
#!/usr/bin/env bash
set -euo pipefail

# Toon hoe vol een LTO-tape is via SCSI Tape Capacity page (0x31).
# Vereist: mt, sg_logs, awk (gawk), grep
#
# Gebruik:
#   sudo ./tapeinfo.sh
#   sudo ./tapeinfo.sh --no-eod
#
# Devices kun je overrulen via env:
#   sudo TAPE_DEV=/dev/nst0 SCSI_DEV=/dev/sg3 ./tapeinfo.sh

TAPE_DEV="${TAPE_DEV:-/dev/nst0}"
SCSI_DEV="${SCSI_DEV:-/dev/sg3}"
DO_EOD=1

if [[ "${1:-}" == "--no-eod" ]]; then
  DO_EOD=0
fi

log() { echo -e "\n== $* =="; }

need_root() {
  if [[ $EUID -ne 0 ]]; then
    echo "Run dit script met sudo/root."
    exit 1
  fi
}

check_cmds() {
  for c in mt sg_logs awk grep; do
    if ! command -v "$c" >/dev/null 2>&1; then
      echo "Mist commando: $c"
      exit 1
    fi
  done
```

```
}

show_status() {
  mt -f "$TAPE_DEV" status 2>/dev/null || true
}

to_eod() {
  if [[ $DO_EOD -eq 1 ]]; then
    log "Positioneer naar EOD (end of data)"
    mt -f "$TAPE_DEV" eod
  else
    log "EOD overslaan (--no-eod)"
  fi
}

read_capacity() {
  sg_logs --page=0x31 "$SCSI_DEV"
}

main() {
  need_root
  check_cmds

  log "Drive/tape status"
  show_status

  to_eod

  log "Tape Capacity log page (0x31)"
  page="$(read_capacity)"

  rem=$(echo "$page" | awk -F': ' '/Main partition remaining capacity/
{print $2; exit}')
  max=$(echo "$page" | awk -F': ' '/Main partition maximum capacity/  {print
$2; exit}')

  if [[ -z "${rem:-}" || -z "${max:-}" || "$max" == "0" ]]; then
    echo "Kon geen geldige capaciteit uitlezen."
    echo "Zorg dat er een tape ONLINE is en probeer opnieuw."
    exit 2
  fi

  used=$(( max - rem ))

  pct_used=$(awk -v u="$used" -v m="$max" 'BEGIN {printf "%.1f",
(u*100.0)/m}')
  pct_free=$(awk -v r="$rem"  -v m="$max" 'BEGIN {printf "%.1f",
(r*100.0)/m}')

  used_gib=$(awk -v u="$used" 'BEGIN {printf "%.1f", u/1024.0}')
  rem_gib=$(awk -v r="$rem"  'BEGIN {printf "%.1f", r/1024.0}')
```

```
  max_gib=$(awk -v m="$max"  'BEGIN {printf "%.1f", m/1024.0}')

  log "Tape capaciteitsrapport"
  echo "Tape device : $TAPE_DEV"
  echo "SCSI device : $SCSI_DEV"
  echo
  echo "Totaal   : $max MiB  (~$max_gib GiB)"
  echo "Gebruikt : $used MiB (~$used_gib GiB)  -> $pct_used% vol"
  echo "Vrij     : $rem MiB  (~$rem_gib GiB)   -> $pct_free% vrij"
}

main "$@"
```

Copy.sh

```
#!/usr/bin/env bash
set -euo pipefail

echo "=== PV copy tool (1 bestand) ==="
echo

have_cmd() { command -v "$1" >/dev/null 2>&1; }

# ----------------------------
# Source file picker (fzf)
# ----------------------------
select_src_file() {
  local start_dir="${1:-$PWD}"

  if have_cmd fzf; then
    # Alleen files, vanaf start_dir
    find "$start_dir" -type f 2>/dev/null \
      | fzf --prompt="Selecteer bronbestand: " --height=80% --reverse
  else
    local src
    read -r -e -p "Bronbestand (volledig pad): " src
    printf "%s" "$src"
  fi
}

# ----------------------------
# Destination directory picker (ONLY dirs, 1 level per keer)
# - browse level-by-level
# - choose parent
# - choose current dir
# ----------------------------
pick_dir_level_by_level() {
  local cur="${1:-$PWD}"

  # Normaliseer pad (resolve symlinks, etc.)
```

```
  cur="$(cd "$cur" 2>/dev/null && pwd -P)" || cur="$PWD"

  while true; do
    local parent choice
    parent="$(dirname "$cur")"

    if have_cmd fzf; then
      choice="$(
        {
          echo "[..] (parent: $parent)"
          echo "[.]  (kies deze map: $cur)"
          # Alleen subdirectories, 1 level diep
          find "$cur" -mindepth 1 -maxdepth 1 -type d -printf "%f/\n"
2>/dev/null | sort
        } | fzf --prompt="Doelfolder: $cur > " --height=80% --reverse
      )" || return 1
    else
      local dst
      read -r -e -p "Doelfolder (pad): " dst
      printf "%s" "$dst"
      return 0
    fi

    case "$choice" in
      "[..]"*)
        cur="$parent"
        ;;
      "[.]"*)
        printf "%s" "$cur"
        return 0
        ;;
      */)
        local sub="${choice%/}"
        cur="$cur/$sub"
        ;;
      *)
        ;;
    esac
  done
}

# ----------------------------
# Mount helpers
# ----------------------------
mountpoint_for_path() {
  # print het mountpoint (TARGET) voor een pad
  findmnt -no TARGET --target "$1" 2>/dev/null || true
}

mount_opts_for_mountpoint() {
  # print mount options voor mountpoint
```

```
  findmnt -no OPTIONS "$1" 2>/dev/null || true
}

is_mount_ro() {
  local mp="$1"
  [[ -n "$mp" ]] || return 1
  mount_opts_for_mountpoint "$mp" | grep -qw ro
}

fstype_for_path() {
  findmnt -no FSTYPE --target "$1" 2>/dev/null || true
}

avail_bytes_on_mount() {
  local path="$1"
  df -B1 --output=avail "$path" | tail -n1 | tr -d ' '
}

human_bytes() {
  # simpele humanizer (B, KiB, MiB, GiB, TiB)
  local b="$1"
  local -a units=("B" "KiB" "MiB" "GiB" "TiB" "PiB")
  local u=0
  local v="$b"
  while [[ "$v" -ge 1024 && "$u" -lt 5 ]]; do
    v=$((v/1024))
    u=$((u+1))
  done
  printf "%s %s" "$v" "${units[$u]}"
}

is_ltfs_path() {
  # Detecteer LTFS op basis van fstype of pad
  local fstype
  fstype="$(fstype_for_path "$1")"
  [[ "$fstype" == "ltfs" ]] && return 0
  [[ "$1" == /mnt/ltfs* ]] && return 0
  return 1
}

# ----------------------------
# 1) Bron selecteren
# ----------------------------
SRC="$(select_src_file "$PWD")" || true

if [[ -z "${SRC:-}" ]]; then
  echo "Geen bron opgegeven, exit." >&2
  exit 1
fi

if [[ ! -f "$SRC" ]]; then
```

```
  echo "Bronbestand bestaat niet: $SRC" >&2
  exit 1
fi

# ----------------------------
# 2) Doelfolder selecteren (1 level per keer)
# ----------------------------
DST_DIR="$(pick_dir_level_by_level "$PWD")" || true

if [[ -z "${DST_DIR:-}" ]]; then
  echo "Geen doelfolder opgegeven, exit." >&2
  exit 1
fi

mkdir -p "$DST_DIR"

DST_MOUNT="$(mountpoint_for_path "$DST_DIR")"
DST_FSTYPE="$(fstype_for_path "$DST_DIR")"

# Safety 1: stop direct als het destination filesystem read-only is
if [[ -n "$DST_MOUNT" ]] && is_mount_ro "$DST_MOUNT"; then
  echo
  echo "ERROR: Doel filesystem is read-only."
  echo "  Doelpad      : $DST_DIR"
  echo "  Mountpoint   : $DST_MOUNT"
  echo "  Fstype       : ${DST_FSTYPE:-unknown}"
  echo
  echo "Dit gebeurt vaak bij LTFS als de data-partitie geen ruimte meer
heeft om een nieuwe index te schrijven."
  exit 1
fi

# ----------------------------
# 3) Doelbestandsnaam
# ----------------------------
DEFAULT_NAME="$(basename "$SRC")"
read -r -e -p "Doelbestandsnaam [${DEFAULT_NAME}]: " DST_NAME
DST_NAME="${DST_NAME:-$DEFAULT_NAME}"

DST="${DST_DIR%/}/$DST_NAME"

# ----------------------------
# 4) Overschrijf-check
# ----------------------------
if [[ -e "$DST" ]]; then
  echo
  echo "Doelbestand bestaat al: $DST"
  read -r -p "Overschrijven? (y/N): " ans
  case "${ans:-N}" in
    y|Y|yes|YES) echo "Oké, ik overschrijf het bestand." ;;
    *) echo "Afgebroken. Geen wijzigingen gedaan."; exit 0 ;;
```

```
  esac
fi

# ----------------------------
# 5) Space-check vooraf
# ----------------------------
SRC_SIZE="$(stat -c%s "$SRC")"
AVAIL="$(avail_bytes_on_mount "$DST_DIR")"

# MARGES:
# - Op normale FS is 1GiB genoeg.
# - Op LTFS: neem liever groter, omdat index writes ruimte nodig hebben.
if is_ltfs_path "$DST_DIR"; then
  SAFETY_MARGIN=$((10*1024*1024*1024))  # 10 GiB marge voor LTFS
(index/metadata + “niet tot 99% vullen”)
else
  SAFETY_MARGIN=$((1*1024*1024*1024))   # 1 GiB voor normale filesystems
fi

NEEDED=$((SRC_SIZE + SAFETY_MARGIN))

if [[ "$AVAIL" -lt "$NEEDED" ]]; then
  echo
  echo "ERROR: Onvoldoende vrije ruimte op doel filesystem."
  echo "  Bronbestand grootte          : $SRC_SIZE bytes ($(human_bytes
"$SRC_SIZE"))"
  echo "  Vrij beschikbaar (df avail)  : $AVAIL bytes ($(human_bytes
"$AVAIL"))"
  echo "  Benodigd incl. marge         : $NEEDED bytes ($(human_bytes
"$NEEDED"))"
  echo
  if is_ltfs_path "$DST_DIR"; then
    echo "LTFS-tip: als de tape bijna vol is, kan LTFS read-only worden
omdat hij geen index meer kan schrijven."
    echo "Kies een nieuwe tape/volume of laat meer vrije ruimte over."
  fi
  exit 1
fi

# ----------------------------
# 6) Kopiëren met pv (veilig: eerst temp, dan mv)
# ----------------------------
echo
echo "Kopieer:"
echo "  Bron : $SRC"
echo "  Doel : $DST"
echo "  (Schrijf eerst naar partial en hernoem bij succes)"
echo

TMP_DST="${DST}.partial.$$"
```

```
cleanup() {
  # Ruim partial op als we falen/stoppen.
  # (Als destination read-only werd vlak voor mv, laten we hem soms bewust
staan)
  if [[ -e "$TMP_DST" ]]; then
    rm -f "$TMP_DST" 2>/dev/null || true
  fi
}
trap cleanup EXIT INT TERM

# Copy
pv -s "$SRC_SIZE" "$SRC" > "$TMP_DST"
sync

# Safety 2: vlak vóór final mv opnieuw RO checken (LTFS kan intussen ro
worden)
DST_MOUNT2="$(mountpoint_for_path "$DST_DIR")"
if [[ -n "$DST_MOUNT2" ]] && is_mount_ro "$DST_MOUNT2"; then
  echo
  echo "ERROR: Doel filesystem is intussen read-only geworden."
  echo "Partial bestand blijft staan voor inspectie:"
  echo "  $TMP_DST"
  echo
  echo "Je kunt het partial bestand handmatig verwijderen of naar een andere
locatie kopiëren."
  trap - EXIT INT TERM
  exit 1
fi

# Rename to final
mv -f "$TMP_DST" "$DST"
trap - EXIT INT TERM

echo
echo "Klaar. Resultaat:"
ls -lh "$DST"
```

tape-age.sh

```
for id in 0x0400 0x0401 0x0406 0x0806 0x0803 0x0003 0x0222 0x0223 0x0220
0x0221; do
  sudo sg_read_attr -f $id /dev/sg4
done
```

From:
https://info.kosternet.nl/ - **KosterNET Info**

Permanent link:
https://info.kosternet.nl/doku.php?id=publiek:ltfs_en_tape&rev=1788844923

Last update: **2026/09/08 07:22**