

Talabalar uchun amaliy topshiriqlar: Fayl tizimlari va Linux buyruqlari

1-Qism: Nazariy savollar

1. **Fayl tizimi** nima va uning asosiy vazifalari (funktsiyalari) nimadan iborat?
2. **Linux** va **Windows** fayl tizimlari (masalan, ext4 va NTFS) o'rtasidagi asosiy farqlarni tushuntirib bering. Ildiz katalogi (root) ikkala tizimda qanday ifodalanadi?
3. **Mutlaq (absolute)** va **nisbiy (relative)** yo'llar o'rtasidagi farq nima? Ikkalasiga ham o'z misolingizni keltiring.
4. **Jurnallashtirish (journaling)** nima va u fayl tizimiga qanday afzallik beradi?
5. Linux tizimlarida **inode** nima va u o'z ichida fayl haqidagi qanday ma'lumotlarni (metama'lumotlarni) saqlaydi?
6. **Qattiq havola (hard link)** va **simvolik havola (symlink)** o'rtasidagi asosiy farqlar nimada?

2-Qism: Amaliy vazifalar (Linux terminalida bajarish uchun)

Ushbu vazifalarni tizimingizdagi Linux terminalida (yoki virtual mashinada) bajaring va har bir qadam uchun foydalangan buyruqlaringizni yozib oling.

1. Kataloglar bilan ishlash:

- Terminalni oching va o'zingizning uy katalogingizga (~) o'ting.
- `os_practice` nomli yangi katalog yarating.
- Yangi yaratilgan katalogning ichida `docs`, `scripts` va `backup` nomli kataloglarni bitta buyruq yordamida yarating.

2. Fayllar yaratish va ko'chirish:

- `docs` katalogiga o'ting va `report.txt` hamda `notes.txt` fayllarini yarating.
- `report.txt` fayli ichiga "Bu operatsion tizimlar fanidan amaliy ish" degan matnni yozing (maslahat: `echo` buyrug'idan foydalanishingiz mumkin).
- `notes.txt` faylini `backup` katalogiga ko'chiring (`mv` buyrug'i yordamida).

3. Fayl huquqlari (Permissions) bilan ishlash:

- `scripts` katalogiga o'ting va bo'sh `run_task.sh` faylini yarating.
- Ushbu faylga faqat fayl egasi (owner) uchun o'qish, yozish va ishga tushirish (`rw`), boshqalar uchun esa faqat o'qish huquqini bering (`chmod` yordamida). Raqamli formatdan (masalan: `chmod 744`) foydalanib ko'ring.
- `ls -l` buyrug'i yordamida huquqlar to'g'ri o'rnatilganligini tekshiring.

4. Fayllarni qidirish va matn tahlili:

- Tizimingizdagi `/var/log` katalogidan barcha `.log` kengaytmali fayllarni topish uchun tegishli `find` buyrug'ini yozing va sinab ko'ring.
- O'zingiz yaratgan `report.txt` fayldan "tizimlar" so'zini qidiruvchi `grep` buyrug'ini yozing.

5. Disk maydoni va tizim ma'lumotlari:

- Tizimingizdagi band va bo'sh disk maydonini odam o'qishi mumkin bo'lgan (human-readable, ya'ni MB, GB) formatda ekranga chiqaring (df).
- O'zingiz yaratgan os_practice katalogi qancha disk xotirasini egallayotganini aniqlang (du).

3-Qism: Murakkabroq vazifa (Ixtiyoriy)

1. docs/report.txt fayli uchun scripts katalogining ichida report_symlink.txt nomli simvolik (yumshoq) havola yarating.
2. docs/report.txt fayli uchun scripts katalogining ichida report_hardlink.txt nomli qattiq havola (hard link) yarating.
3. Asl faylni (docs/report.txt) o'chirib tashlang.
4. Endi cat buyrug'i yordamida report_symlink.txt va report_hardlink.txt fayllarini o'qishga harakat qilib ko'ring.
5. Nima yuz berganini va nima uchun shunday bo'lganini tushuntiring.

Javoblar (Solutions)

1-Qism: Nazariy savollar javoblari

1. **Fayl tizimi** — tashuvchida fayllar va kataloglarni tashkil etish, saqlash va boshqarish mexanizmi. Uning asosiy vazifalari: ma'lumotlarni saqlash, topish, ruxsatlarni boshqarish, bo'sh joyni nazorat qilish va ma'lumotlar butunligini ta'minlash.
2. Linuxda hamma narsa yagona ierarxiyada (/ ildiz katalogida) birlashadi. Windowsda esa har bir bo'lim alohida mantiqiy disklarga (masalan, C:\) ega.
3. **Mutlaq yo'l**: Fayl tizimi ildizidan boshlanadigan to'liq manzil (masalan, /home/user/docs/report.txt). **Nisbiy yo'l**: Joriy turgan katalogga nisbatan hisoblangan yo'l (masalan, joriy katalog /home/user/ bo'lsa, nisbiy yo'l docs/report.txt bo'ladi).
4. **Jurnallashtirish**: O'zgarishlarni diskka yozishdan oldin maxsus jurnalga yozib qo'yish mexanizmi. Bu kompyuter kutilmaganda o'chib qolsa, ma'lumotlarni tezroq va xatosiz tiklashga yordam beradi.
5. **Inode** faylning metama'lumotlarini (o'lchami, egasi, ruxsatlari, yaratilgan vaqti va diskdagi jismoniy joylashuvi manzillarini) saqlaydi. U fayl nomini o'z ichiga olmaydi.
6. **Simvolik havola (Symlink)** asl faylning yo'lga (manziliga) ishora qiladi. Asl fayl o'chirilsa, havola ishlamay qoladi (buziladi). **Qattiq havola (Hard link)** esa bevosita ma'lumotlar blokiga (inodega) ishora qiladi. Asl fayl o'chirilsa ham, qattiq havola orqali ma'lumotlarni o'qish mumkin.

Qattiq havola (hard link) va simvolik havola (symlink) o'rtasidagi farqni tushunish uchun quyidagi amaliy misollarni terminalda bajarib ko'rishingiz mumkin:

1. Dastlabki tayyorgarlik va fayl yaratish

Birinchi navbatda, sinov uchun alohida papka va yangi fayl yaratib olamiz:

```
mkdir link_test
cd link_test
echo "Bu mening asl ma'lumotim" > original.txt
```

2. Havolalarni (links) yaratish

Endi ushbu fayl uchun bitta qattiq havola (hard link) va bitta simvolik (yumshoq) havola (symlink) yaratamiz:

- **Qattiq havola yaratish:** In buyrug'idan paramsiz foydalaniladi.

```
ln original.txt hardlink.txt
```

- **Simvolik havola yaratish:** In buyrug'iga -s (symbolic) parametri qo'shiladi.

```
ln -s original.txt symlink.txt
```

3. Inode raqamlarini tekshirish

Fayllarning qanday bog'langanligini ko'rish uchun ularning Inode (indeks tuguni) raqamlarini tekshiramiz:

```
ls -li
```

Natijada shunga o'xshash yozuv chiqadi:

```
1234567 -rw-r--r-- 2 user user 25 Oct  2 09:10 hardlink.txt
1234567 -rw-r--r-- 2 user user 25 Oct  2 09:10 original.txt
1234568 lrwxrwxrwx 1 user user 12 Oct  2 09:10 symlink.txt -> original.txt
```

Tahlil: E'tibor qilsangiz, original.txt va hardlink.txt fayllarining Inode raqamlari (eng birinchi ustundagi son, masalan 1234567) bir xil. Chunki qattiq havola to'g'ridan-to'g'ri asl fayl joylashgan xotira qismiga ishora qiladi. symlink.txt esa boshqa Inode raqamiga ega (1234568) va u faqat faylning nomiga (yo'liga) havola qilmoqda (-> original.txt).

4. Asl faylni o'zgartirib ko'rish

Agar asl faylni o'zgartirsak, ikkala havola orqali o'qiganda ham o'zgarishlar ko'rinadi:

```
echo "Yangi qator qo'shildi" >> original.txt

cat hardlink.txt
cat symlink.txt
```

Ikkalasidan ham bir xil natija chiqadi.

5. Asl faylni o'chirish (Asosiy farq shu yerda ko'rinadi)

Endi original.txt (asl fayl) ni o'chirib tashlaymiz:

```
rm original.txt
```

Endi havolalarni o'qishga harakat qilamiz:

- **Qattiq havolani o'qish:**

```
cat hardlink.txt
```

Natija: Fayl ichidagi matn muvaffaqiyatli o'qiladi. Chunki qattiq havola to'g'ridan-to'g'ri diskdagi ma'lumotlarga ulangan.

- **Simvolik havolani o'qish:**

```
cat symlink.txt
```

Natija: cat: symlink.txt: No such file or directory (Bunday fayl yoki katalog yo'q). *Chunki simvolik havola faqat original.txt nomini ko'rsatayotgan edi, asl fayl o'chib ketgach, havola ishlamay qoladi (broken link).*

2-Qism: Amaliy vazifalar javoblari (Buyruqlar)

1. Kataloglar bilan ishlash:

```
cd ~
mkdir os_practice
cd os_practice
mkdir docs scripts backup
```

2. Fayllar yaratish va ko'chirish:

```
cd docs
touch report.txt notes.txt
echo "Bu operatsion tizimlar fanidan amaliy ish" > report.txt
mv notes.txt ../backup/
```

3. Fayl huquqlari (Permissions):

```
cd ../scripts
touch run_task.sh
chmod 744 run_task.sh
ls -l
```

4. Fayllarni qidirish va tahlil qilish:

```
find /var/log -name "*.log"
grep "tizimlar" ../docs/report.txt
```

5. Disk maydoni:

```
df -h
du -sh ~/os_practice
```

3-Qism: Murakkabroq vazifa javobi

1. Simvolik havola: `ln -s ../docs/report.txt report_symlink.txt`
2. Qattiq havola: `ln ../docs/report.txt report_hardlink.txt`
3. Asl faylni o'chirish: `rm ../docs/report.txt`
4. O'qishga urinish:
 - `cat report_symlink.txt` (Xatolik beradi: No such file or directory)
 - `cat report_hardlink.txt` (Muvaffaqiyatli o'qiladi)
5. **Sababi:** Simvolik havola faqat fayl yo'lini ko'rsatgan edi, asl fayl o'chgach u yo'l bo'shab qoldi (broken link). Qattiq havola esa bevosita inodega ulangan edi, shuning uchun ma'lumotlar fizik diskdan o'chmadi va qattiq havola ushbu ma'lumotlarga kirishni ta'minlab qoldi.

Qattiq (hard link) va simvolik (symlink - yumshoq) havolalar operatsion tizimlarni (ayniqsa, Linux/Unix tizimlarini) boshqarishda juda muhim ahamiyatga ega. Ularning har biri o'ziga xos muammolarni hal qilish uchun ishlatiladi.

Quyida ularning real amaliyotdagi (production va tizim administratsiyasidagi) jiddiy misollarini keltirib o'tamiz.

1. Simvolik havolalar (Symlink) ning amaliyotdagi o'rni

Simvolik havolalar asosan **versiyalarni boshqarish, konfiguratsiyalarni yoqish/o'chirish va qulaylik** yaratish uchun ishlatiladi. Ular disk chegaralarini buzib o'ta oladi (ya'ni boshqa qattiq disklarga ham havola bera oladi) va papkalar (kataloglar) uchun ham ishlaydi.

Jiddiy amaliy misollar:

- **1. Dastur versiyalarini boshqarish (Python, Node.js, Java):**

Aytaylik, serveringizda Python'ning bir nechta versiyasi bor (python3.8, python3.10). Tizim va boshqa dasturlar doim python buyrug'i orqali ishlashi uchun simvolik havola ishlatiladi:

```
/usr/bin/python -> /usr/bin/python3.10
```

Agar ertaga python3.11 ga o'tmoqchi bo'lsangiz, barcha dasturlarning kodini o'zgartirib chiqmaysiz, shunchaki bitta simvolik havolani yangilab qo'yasiz xolos. Bu juda mashhur amaliyot (masalan, Debian/Ubuntu dagi update-alternatives xuddi shunday ishlaydi).

- **2. Veb-server konfiguratsiyalarini boshqarish (Nginx, Apache):**

Nginx veb-serverida saytlar sozlamalari sites-available (saytlar mavjud) papkasida saqlanadi. Saytni internetda faollashtirish uchun uni to'g'ridan-to'g'ri ko'chirmaysiz, balki sites-enabled (saytlar yoqilgan) papkasiga simvolik havola qilasiz:

```
ln -s /etc/nginx/sites-available/mening_saytim.conf /etc/nginx/sites-enabled/
```

Saytni vaqtincha o'chirish kerak bo'lsa, faqat havolani o'chirasiz (rm /etc/nginx/sites-enabled/mening_saytim.conf). Asl sozlama saqlanib qoladi.

- **3. Umumiy kutubxonalar (Shared Libraries - .so fayllari):**

Linuxda dasturlar ishlaydigan libcrypto.so kabi fayllar aslida qaysidir aniq versiyaga ishora qiluvchi simvolik havolalardir.

```
libcrypto.so -> libcrypto.so.1.1
```

Kutubxona yangilanganda (masalan, xavfsizlik yangilanishi), asl fayl libcrypto.so.1.2 ga almashadi va havola unga burib qo'yiladi. Dasturlar esa hamon libcrypto.so ni qidiradi va uzluksiz ishlayveradi.

2. Qattiq havolalar (Hard link) ning amaliyotdagi o'rni

Qattiq havolalar asosan **disk joyini tejash, ma'lumotlar xavfsizligini ta'minlash va zaxira nusxalarini (backups)** yaratishda ishlatiladi. Ular faqat bitta disk (fayl tizimi) ichida ishlaydi va papkalar uchun yaratilmaydi.

Jiddiy amaliy misollar:

- **1. Zaxira nusxalarini (Backups) optimallashtirish (Eng yirik amaliyot):**

Tasavvur qiling, serveringizda 1 Terabayt ma'lumot bor va siz uni har kuni zaxiralaysiz (backup qilasiz). Agar har kuni buni oddiy nusxalasangiz, diskda juda tez joy qolmaydi.

rsnapshot yoki Apple'ning Time Machine kabi professional backup dasturlari **hard link** dan foydalanadi.

Agar kechadan beri fayl o'zgarmagan bo'lsa, tizim uni qayta nusxalamaydi, balki oldingi kungi zaxira nusxasiga *qattiq havola* yaratib qo'yadi xolos. Natijada sizda butun boshli "Bugungi zaxira nusxasi" papkasi ko'rinadi, lekin fayllar diskda jismoniy jihatdan faqat 1 marta joy egallaydi. Eski kunning zaxirasini o'chirib tashlasangiz ham, fayl jismoniy o'chmaydi, chunki bugungi kundagi hard link uni ushlab turadi.

- **2. Katta hajmli fayllarni xavfsiz almashish:**

Sizda 50 GB lik ma'lumotlar bazasi (Database dump) yoki videofayl bor. Uni boshqa foydalanuvchiga (boshqa papkaga) berishingiz kerak. Agar nusxa olsangiz (copy), u diskdan yana 50 GB joy oladi va nusxalash uzoq vaqt oladi.

Buning o'rniga qattiq havola qilasiz:

```
ln /var/data/big_database.sql /home/dev/big_database.sql
```

Bu jarayon 1 soniyadan ham kam vaqt oladi va diskdan qo'shimcha 0 bayt joy oladi. Ikkala foydalanuvchi ham xuddi o'zida alohida fayl bordek ishlayveradi. Eng muhimi, bir foydalanuvchi o'z faylini bilmasdan o'chirib yuborsa ham, ikkinchi havola tirik bo'lgani uchun fayl yo'qolib ketmaydi.

- **3. Git kabi versiya nazorat tizimlari (VCS):**

Katta Git repozitoriyni bitta kompyuter ichida boshqa papkaga git clone qilganingizda, Git joyni tejash uchun nusxa ko'chirish o'rniga tez-tez qattiq havolalardan foydalanadi (local clone holatida).

Qisqacha xulosa:

- **Symlink** — bu qulaylik, egiluvchanlik va tezkor boshqaruv (versiyalar, yo'llar, server sozlamalari). Bu xuddi Windows'dagi "Yarlik" (Shortcut) ga o'xshaydi, faqat undan ancha aqlliyoq.
- **Hard link** — bu gigabaytlik disk joyini tejash tezkor nusxalash va fayl himoyasi. U tizimga bitta fizik faylni "Men turli papkalardagi ikkita mustaqil faylman" deb aldamchi ko'rsatish imkonini beradi.