



# Операційні системи

Лекція 15

Розподілені файлові системи



# План лекції



# Функції розподілених файлових систем

- Функції розподілених файлових систем такі ж, як і в централізованих системах
  - Зберігання програм і даних
  - Надання доступу до них



# Файл-сервери

- Файлова система підтримується *файл-серверами*
- Файл-сервери:
  - перехоплюють запити на зчитування або записування
  - перевіряють їх
  - виконують їх
  - відповідь надсилають відправнику запиту
- Зазвичай файл-сервери мають ієрархічні файлові системи
- Клієнти (робочі станції) можуть монтувати ці файлові системи до своїх локальних файлових систем



# Файловий сервіс vs файловий сервер

- *Файловий сервіс* – опис функцій, які файлова система пропонує користувачам
  - Примітиви, що існують
  - Їхні параметри
  - Функції, що вони виконують
- Фактично файловий сервіс – це інтерфейс файлової системи з клієнтами
  - Тобто, з точки зору користувача описано, **що** можна зробити, але не описано, **як** це реалізовано



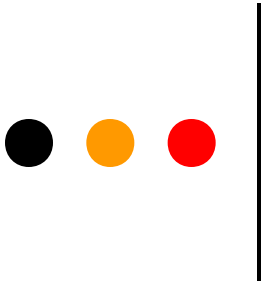
# Файловий сервіс *vs* файловий сервер

- *Файловий сервер* – це процес, що виконується на окремій машині і забезпечує реалізацію файлового сервісу
- У добре організованій розподіленій системі користувачі не знають кількості файлових серверів і їхнього місцезнаходження
- *Сервіс прозорий*
- Зазвичай файловий сервер – це просто процес (режиму користувача або ядра)
- У системі можуть бути присутніми кілька файлових серверів, що реалізують різні функції
  - Наприклад, файлові сервіси Windows і Unix



# Інтерфейс файлового сервісу

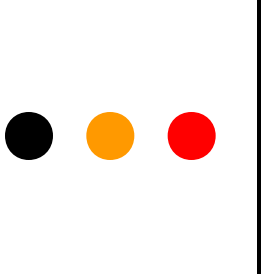
- Перше питання – як інтерпретувати файл (як послідовність байт або записів)
  - У сучасних розподілених системах, як і в централізованих, - **неінтерпретована послідовність байтів**
- Атрибути файлу
  - Ім'я, розмір, дати, ідентифікатор власника
- Можливість модифікації файлу
  - У більшості систем, але не в усіх!
  - Інший варіант – незмінні файли
    - Реалізують лише дві операції – **створити** файл і **прочитати** файл
    - Легше реалізувати кешування і реплікацію файлу
- Два типи файлового сервісу
  1. **Модель завантаження-вивантаження**
  2. **Модель віддаленого доступу**



# Модель завантаження- вивантаження

- Реалізовані зчитування і записування файлу цілком
- Схема оброблення
  - Зчитування файлу з сервера на машину клієнта
  - Оброблення файлу на машині клієнта
  - Записування оновленого файлу на сервер
- Переваги
  - Концептуальна простота
  - Добре працює, коли потрібна робота з цілим файлом
- Недоліки
  - Високі вимоги до дисків клієнтів
  - Неефективно, коли потрібна лише мала частина файлу





# Модель віддаленого доступу

- Операції над файлами:
  - Відкриття й закриття файлів
  - Зчитування й записування частин файлу
  - Позиціонування у файлі
  - Перевірка і зміна атрибутів файлу
- Уся файлова система виконується на серверах



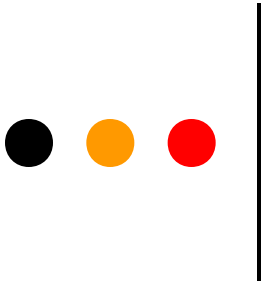
# Сервіс каталогів

- Призначення сервісу – пошук файлу у розподіленій системі
- Найголовніший принцип – забезпечення прозорості
  - 1. Прозорість розміщення
    - Імена файлів не дають можливості з'ясувати, де розміщено файл
    - Наприклад, `/server1/dir1/dir2/x` – невідомо, де знаходиться сервер
    - Якщо перенести `server1` з одної машини на іншу, система збереже працездатність
  - 2. Незалежність від розміщення
    - Ім'я файлу не містить у явному вигляді посилання на будь-який сервер
    - Система може автоматично перенести файл з одного сервера на інший
    - Якщо файлова система базується на принципі віддаленого монтування – вона не забезпечує незалежності від розміщення



# Копії файлу

- У централізованих системах існують унікальні імена файлів
- У розподілених системах може бути, що унікальне ім'я відповідає кільком копіям файлу
  - Це підвищує відмовостійкість за рахунок надлишковості



# Проблема спільного користування файлами

- Необхідно точно визначити семантику зчитування і записування
  1. Семантика UNIX
  2. Сесійна семантика
  3. Незмінні файли
  4. Неподільні транзакції



# Семантика UNIX

- Якщо операція зчитування здійснюється після операції записування, то зчитують вже оновлений файл
- Якщо було дві операції записування, то зчитується результат останньої операції
- У централізованій системі легко і зрозуміти, і реалізувати
- У розподіленій системі можна реалізувати лише якщо є лише один файл-сервер і клієнт не кешує файли
  - Продуктивність значно знижується
- Іноді дозволяють клієнтам кешувати файли, але усі зміни одразу надсилають серверу
  - Це неефективно



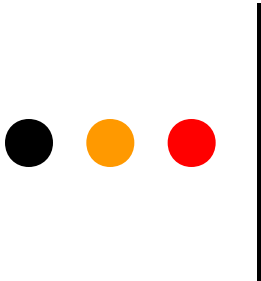
# Сесійна семантика

- Зміни у файлі видимі лише тому процесу, що відкрив цей файл для модифікації
- Усі інші бачать зміни у файлі лише після його закриття
- Існує проблема одночасного використання файлу двома і більше клієнтами
  - Варіант 1 – остаточною є файл, який був закритий останнім
  - Варіант 2 – будь-який з відкритих файлів (простіше реалізувати)



# Структура файлової системи

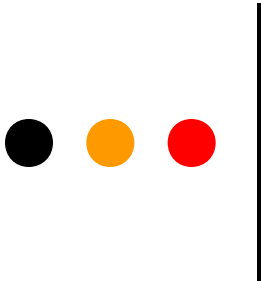
1. Розподіл серверної і клієнтської частин між машинами
2. Структуризація сервісів файлів і каталогів
3. Зберігання на серверах інформації про клієнтів



# Розподіл серверної і клієнтської частин між машинами: варіанти

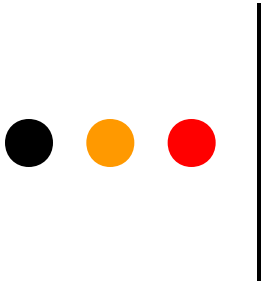
1. Немає жодної різниці між клієнтом і сервером
  - На усіх машинах – одне й те саме базове програмне забезпечення
  - Приклад – NFS
2. Файл-сервер – програма режиму користувача
  - Система може бути сконфігурована як клієнт, як сервер, як клієнт і сервер одночасно
3. Клієнт і сервер – принципово різні машини як у термінах апаратури, так і у термінах програмного забезпечення





# Структуризація сервісів файлів і каталогів: варіанти

1. 2 сервіси на одному сервері
2. Різні машини
  - Цей варіант більш гнучкий
  - Крім того, таким чином можна досягти спрощення ПЗ
  - Недолік – збільшення інтенсивності мережного обміну



# Зберігання на серверах інформації про клієнтів

1. Не зберігати (*stateless*)
2. Зберігати (*statefull*)
  - Statefull сервер пам'ятає, які файли відкрив кожний користувач, положення покажчиків, тощо
  - У разі відмови сервера таблиці втрачаються



# Переваги statefull і stateless серверів

## Stateless

- Відмовостійкість
- Не потрібні виклики OPEN/CLOSE
- Менше пам'яті сервера витрачається
- Немає обмежень на число відкритих файлів
- Відмова клієнта не створює проблем для сервера

## Statefull

- Коротші повідомлення під час запитів
- Краща продуктивність
- Можливе зчитування з випередженням
- Легше досягти ідемпотентності
- Можливе блокування файлів



# Кешування

Диск сервера → пам'ять сервера → пам'ять клієнта → диск клієнта

- Кешування на сервері
  - 1. Якими одиницями оперує кеш
    - Цілі файли – ефективніше зберігання на диску (менше число обмінів)
    - Дискові блоки – ефективніше використання пам'яті кешу і дискового простору
  - 2. Правило заміни даних у кеші
    - Наприклад, алгоритм **LRU** (*Last Recently Used*)
  - Кешування на сервері легко реалізується і прозоро для клієнта
- Кешування на боці клієнта
  - Позбавляє від зайвого трафіку
  - Породжує безліч проблем!



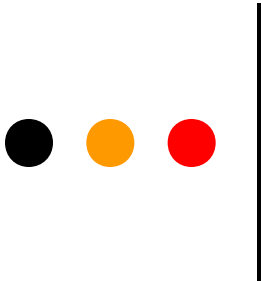
# Рішення проблеми узгодження

1. Алгоритм наскрізного записування
  - Під час модифікації даних у кеші нове значення негайно надсилається серверу
  - Недолік – інтенсивність мережного обміну зменшується лише при зчитуванні
2. Відкладене записування
  - Клієнт помічає, що файл вже змінений
  - Приблизно один раз за 30 секунд усі зміни збираються разом і надсилаються
3. Записування-по-закриттю
  - Сесійна семантика
4. Алгоритм централізованого керування
  - Семантика UNIX
  - Передбачає statefull – підхід
  - Якщо файл вже кимось відкритий, його не можна відкрити для записування



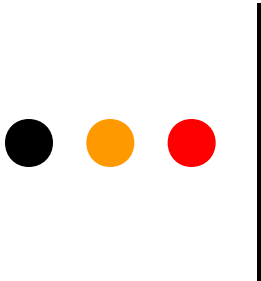
# Реплікація

- Система оперує кількома копіями файлів, причому кожна копія знаходиться на окремому сервері
- Переваги:
  - Підвищення надійності
  - Розподіл навантаження між кількома серверами



# Способи досягнення прозорості реплікації

1. Програміст, що створює прикладну програму, сам керує реплікацією
  - Під час створення файлу автоматично створюються його копії
    - Наприклад,  
/machine1/usr/ast/xyz – основний файл  
/machine2/usr/ast/xyz і  
/machine3/usr/ast/xyz – його копії (записують одночасно)
  - Для розподілених систем такий підхід не рекомендують!
2. “Ледаща” реплікація
  - Спочатку записують файл, а потім роблять його копії
3. Групові зв’язки
  - Визначені групи
  - Файл записують у групу, а система автоматично (одночасно) створює його копії



# Контроль змін реплікованих файлів

1. Реплікація першої копії
  - Виділяють первинний сервер
  - Зміни надсилають на первинний сервер, а він – на вторинні
2. Голосування
  - Відстежують версії файлів
  - Нехай є  $N$  серверів з копіями
  - Під час записування зміни вносять щонайменше у  $W$  копій
  - Під час зчитування переглядають щонайменше  $R$  копій
  - Якщо  $R+W > N$ , то буде знайдена хоча б одна з копій останньої версії
  - Оскільки зчитування здійснюють частіше, ніж записування, то обирають  $R < W$