УДК 004.75

ПОБУДОВА ВИСОКОНАДІЙНОЇ РОЗПОДІЛЕНОЇ ФАЙЛОВОЇ СИСТЕМИ ДЛЯ СІМЕЙСТВА ПЛАНУВАЛЬНИКІВ ЗАДАЧ MAUI

Оніщук А.А.
Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського», Україна, Київ

У роботі розглянуто спосіб впровадження у планувальник задач для GRID систем MAUI концепції розподілених файлових систем для вирішення проблем низької масштабованості сховища даних, неможливості надійного зберігання даних та безперебійного доступу до них через можливі збої у програмній чи апаратній частині обладнання. Запропоновано архітектуру розподіленої файлової системи, яка є підходящою під GRID систем під керівництвом MAUI, а також інших GRID систем, що мають однорідну структуру планувальника задач та працюють по принципу розподілення навантаження між вузлами GRID системи.

Ключові слова: розподілена файлова система, планувальник задач MAUI, масштабованість файлових систем, висока надійність файлових систем, вузол імен, вузол даних

Онищук А.А. Построение высоконадежной распределенной файловой системы для семейства планировщиков задач MAUI / Национальный технический университет Украины «Киевский политехнический институт имени Игоря Сикорского», Украина, Киев

В работе рассмотрен способ внедрения в планировщик задач для GRID систем MAUI концепции распределенных файловых систем для решения проблем низкой масштабируемости хранилища данных, невозможности надежного хранения данных и бесперебойного доступа к нему из-за возможных сбоев в программной или аппаратной части оборудования. Предложена архитектура распределенной файловой системы, которая является подходящей под GRID систем под руководством MAUI, а также других GRID систем, имеющих однородную структуру планировщика задач и работающих по принципу распределения нагрузки между узлами GRID системы.

Ключевые слова: распределенная файловая система, планировщик задач MAUI, масштабируемость файловых систем, высокая надежность файловых систем, узел имен, узел данных.

Onischuk A.A. Creating Highly Available Distributed File System for Maui Family Job Schedulers / National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute", Ukraine, Kyiv

This article describes a way to implement a distributed file system for MAUI job scheduler, which solves the problems of low scalability and non-reliability of data storage, as well as a problem of problem of data inaccessibility due to failures in software or hardware. The architecture which is suitable for MAUI GRID systems is suggested as well as for other GRID systems, which have a uniform structure of job scheduler.

Keywords: distributed file system, MAUI task scheduler, file system scalability, high reliability of file systems, name node, data node.

Вступ. В зв'язку із швидким збільшенням популярності розподілених систем через їх більшу надійність, масштабованість та потужність різко зросла потреба в простому і зручному ПЗ, що спрощувало б роботу користувачів таких систем.

На даний момент користувачу GRID системи для того, щоб запустити задачу, яка працює із файлами у планувальнику Maui потрібно вручну копіювати файли на кожен із вузлів у кластері, що збільшує ймовірність помилок. А також має ряд інших недоліків у порівнянні із використанням концепції розподіленої файлової системи для організації роботи із файлами. Перевагами використанням такого концепту є масштабованість, надійність зберігання даних, надійність доступу до даних, а також дешевизна обладнання для зберігання файлів.

Архітектура розподіленої файлової системи для Maui. Розподілені файлові системи (РФС) – це файлові системи у яких файлові частини (у більшості ФРС блоки) зберігаються на різних носіях з`єднаних високошвидкісною мережею [1]. Система, що описується, для Maui планувальника є підвидом РФС.

У системі є два основні типи вузлів: вузол імен і вузол даних. А два допоміжні типи вузлів: журнальний вузол та резервний вузол імен (будуть розглянуті у наступному розділі). На рис.1 показана архітектура розміщення та взаємодії даних вузлів.

Вузол імен є центральним вузлом РФС. Існує лише один активний вузол імен у РФС. На ньому зберігаються метадані файлової системи, а також інформація про те, де в кластері зберігаються дані файлу. До метаданих належать: імена файлів, їх типи, права доступу, дані розміщення блоків у межах мережі. Вузол імен не зберігає ніяких файлових даних. Це зроблено для того, щоб знизити навантаження на нього. Адже при більшості файлових операцій перший, а іноді і єдиний виклик здійснюється до вузла імен. Виключенням є операція запису у файл, яка потребує координації між усіма типами вузлів файлової системи.

Задачі MAUI щоразу звертаються до вузла імен коли вони хочуть знайти файл, або додати / копіювати / перемістити його. При

читанні файлів вузол імен відповідає на запити поверненням списку відповідних вузлів імен де зберігаються блоки цих даних.

Вузол даних - це клієнтський вузол, основним призначення якого є збереження блоків даних. Для використання переваг РФС у ній має бути більше ніж один вузол даних. Кожен такий вузол знає про блоки, що на ньому знаходяться, доступ до яких користувачеві дає вузол імен за допомогою спеціального посилання, після відповідного запиту. Крім цього, вузли даних вміють робити реплікацію файлових блоків, для підвищення надійності системи. Також для таких вузлів зазвичай не потрібно використання RAID дисків, адже існує реплікація між вузлами.

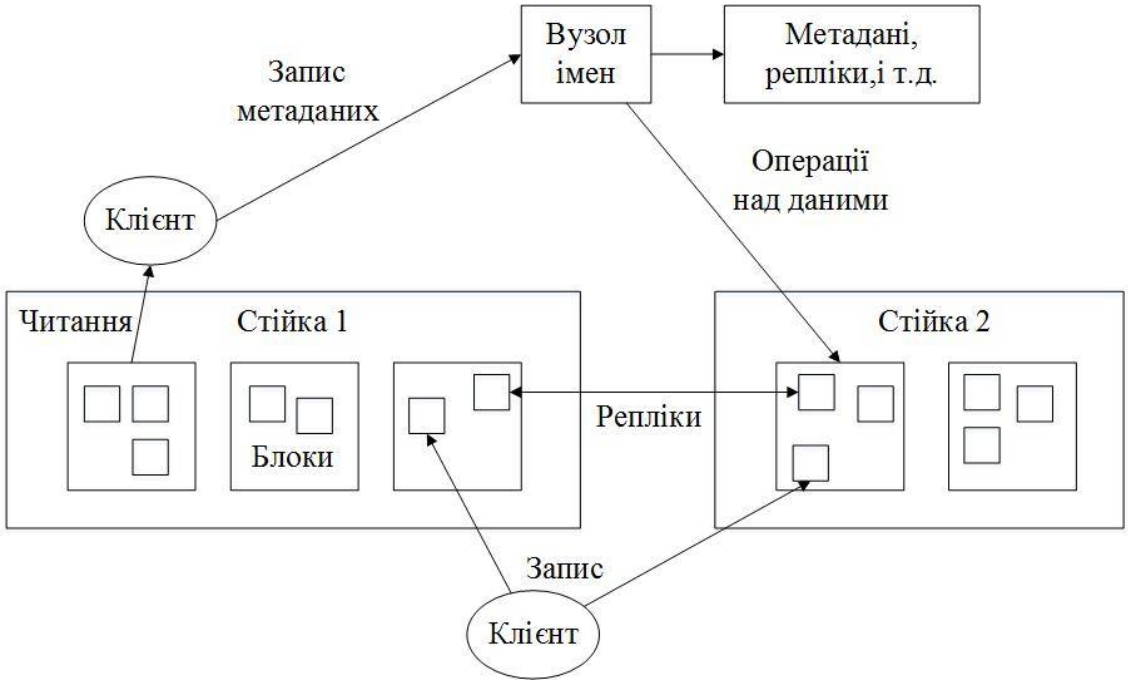

Рис. 1. Архітектура РФС

Для пришвидшення доступу до файлових блоків, вузли імен кешують блоки, які найчастіше використовуються. Тому збільшення кількості RAM на вузлах даних хоча й може призвести до дещо швидшої роботи, проте не є критичним.

Ще одною частиною файлової системи є РФС клієнт. РФС клієнт це програмна бібліотека, яка дозволяє працювати із файловою системою за допомогою простих команд unix інтерфейсу. Дозволені такі команди як: ls, rm, mkdir, touch, а також деякі допоміжні команди притаманні розподіленим файловим системам: copyFromLocal, copyToLocal. Ці команди дозволяють копіювати файли чи папки із локальної файлової системи у РФС і навпаки. Ще один РФС клієнт – MAUI РФС клієнт інтегрується з API інтерфейсом MAUI, та

використовується ним для проведення файлових операцій через вбудований MAUI інтерфейс [2].

Розглянемо на прикладі основних файлових операції взаємодію даних вузлів. Найбільш складною операцією з точки зору взаємодії компонент РФС є операція запису у файл (WRITE). Вона складається із чотирьох кроків описаних нижче:

1. Виклик до вузла імен для отримання списку вузлів даних між якими потрібно розподілити блоки файлу.
2. Запис файлових блоків до вузлів даних.
3. Реплікація вузлами даних отриманих блоків.
4. Надання вузлу імен інформації про розміщення записаних блоків

Операція редагування атрибутів файлу(CHANGEATTR) таких як прав, імені, розміщення обмежується одним зверненням до вузла імен. Теж саме стосується й операцій видалення файлу (RMFILE), створення папки (MKDIR) або пустого файлу (TOUCH). Операція копіювання файлу реалізовується через серію доручень від вузла імен до вузлів даних, для копіювання блоків файлу.

Роль вузла імен та вузла даних як правило виконує типічний комп`ютер економ класу [3]. Це зумовлюється наявністю реплікацією даних і високою надійністю. Тобто, при відмові одного із комп'ютерів, що входить у РФС, не відбувається втрата даних, чи відмова у обслуговуванні. Це сильно здешевлює початкову ціну файлової системи, так як вона може складатись всього із кількох комп'ютерів економ класу і при цьому надає значну масштабованість, через те, що для збільшення потужностей достатньо додати нові вузли у РФС. На відміну від традиційний хранилищ даних, де для цього необхідна заміна обладнання.

Розглянемо концепцію блоку РФС. Звичні нам дискові файлові системи мають розмір блоку, що позначає мінімальну кількістю даних, які він може читати і писати.

Файлові системи для одного диска будуються на цьому принципі роботи з розміром даних в блоку, який кратний розміру блоку на диску. Блоки такий типічний файлових систем, як правило займають кілька кілобайт, в той час як дискові блоки, як правило, 512 байт [4].

РФС теж має концепцію блоку, але це набагато більше, блок має мати значний розмір наприклад 128 МБ.

Блоки РФС такі великі в порівнянні з дисковими блоками, і причина полягає в мінімізації витрат на пошук. Якщо блок досить великий, то час, необхідний для передачі даних з диска може бути значно більшим, ніж час, щоб звернутися до початку блоку. Таким чином, передача великого файлу з декількох блоків, працює зі швидкістю передачі диска.

Швидкий розрахунок показує, що якщо час пошуку складає близько 10 мс, а швидкість передачі 100 МБ / с, то щоб зробити час пошуку 1% від часу передачі, нам потрібно, щоб розмір блоку був близько 100 МБ.

Організація високої надійності для розподіленої файлової системи. Одною із головних переваг розподіленої файлової системи є можливість організації високої надійності зберігання файлів та доступу до них. Для того, щоб досягти цього для початку розглянемо проблеми, що можуть завадити доступу до даних або пошкодити самі дані:

- апаратний чи програмний збій вузла імен
- апаратний чи програмний збій одного або кількох вузлів даних
- недоступність даних через неполадки в мережі

Апаратний чи програмний збій вузла даних вирішується за допомогою реплікації файлових блоків між цими вузлами. Реплікацією у даному сенсі є надлишкове копіювання блоків даних між вузлами даних. Число вузлів на яких зберігаються блоки даних називається фактором реплікації. За допомогою налаштування високого фактора реплікації, а також налаштування реплікації у різні сегменти мережі можна добитись безперервного доступу до файлів у випадках відмови кількох вузлів одразу, або навіть цілих сегментів мережі.

Для того щоб могти забезпечити безперебійну роботу MAUI GRID системи при збої вузла імен потрібне запровадження двох допоміжних типів вузлів: журнального вузла, а також резервного вузла імен.

Дві окремі машини налаштовуються як вузли імен. У будь-який момент часу, рівно один з вузлів імен знаходиться в активному стані, а інший знаходиться в стані очікування (резервний вузол імен). Активний вузол імен відповідає за всі клієнтські операції в кластері, в той час як резервний є в режим очікування та не використовується, при цьому він зберігає достатньо інформації про стан файлової системи, щоб забезпечити швидкий перехід на нього, якщо це необхідно.

Для того, щоб резервний вузол імен був синхронізованим з активним вузлом, обидва вузла зв'язуються з групою окремих вузлів, так званих, журнальних вузлів. Коли будь-яка зміна імен виконуються активним вузлом, він реєструє запис про модифікацію для більшості журнальних вузлів. Резервний вузол імен здатний читати правки з журнальних вузлів і постійно спостерігає за змінами у їхньому журнал. Як резервний вузол бачить зміни, він застосовує їх до свого власного простору імен. У разі відмови активного вузла імен, резервний, після читання всіх правок з журналу імен, оголошує себе активним. Це

гарантує, що стан простору імен повністю синхронізований, перш ніж відбувається перехід на інший вузол.

Для того, щоб забезпечити швидкий перехід простору імен на інший ресурс необхідно, щоб резервний вузол імен мав останню актуальну інформацію про місцезнаходження блоків в кластері. З цією цілю, вузли даних передають інформацію про місцезнаходження блоків, а також поточний свій статус обом вузлам імен.

Дуже важливим для правильної роботи такої системи, є той факт що тільки один з вузлів імен має бути активними в кожен момент часу. В іншому випадку стан простору імен швидко розходяться між ними, що приводить до втрати даних або інших невірних результатів. Для того, щоб забезпечити цю властивість і запобігти так званому «сценарій розколу мозку», журнальні вузли за весь час головування вузла імен дозволяють запис подій лише йому. Під час переходу на інший ресурс простору імен, новий активний вузол імен просто бере на себе обов'язок запису в журнальні вузли, що буде ефективно запобігати іншому вузлу імен продовжувати бути в активному стані.

Недоступність ж вузлів через неполадки мережі може бути легко усунена за допомогою організації надлишкової топології мережі.

В основі концепції надлишкової топології мережі лежить потреба забезпечення альтернативних шляхів для даних, щоб переміщатися в разі, якщо кабель пошкоджений або роз'їм не підключений [5]. Проте, Ethernet як стандарт, не допускає кільця або петлі в мережі, так як це викликає широкомовні цикли і в кінцевому підсумку призводить до того що мережа призупиняє роботу. Мережа Ethernet не може мати два шляхи з точки А в точку Б без механізму для підтримки цього типу топології. Для забезпечення надмірності, мережева інфраструктура (комутатори) повинні підтримувати протоколи резервування, покликані звести нанівець проблему введення петлі в мережу Ethernet. Це досягається шляхом зберігання кількох альтернативних шляхів на комутаторі: шляху до даних за замовчуванням і альтернативного шляху, які він перемикає коли виникає потреба.

Висновки. У даній роботі було описано архітектуру розподіленої файлової системи для MAUI, яка дозволяє досягнути більшої масштабованості, а також досягнути високої надійності зберігання даних, та їх доступності.

Література:

1. Ilya Ganelin. Spark: Big Data Cluster Computing in Production / Ilya Ganelin // Wiley, March 2016. cm.21-32.

2. Сайт Maui [Електронний ресурс] // Режим доступу: http://www.adaptivecomputing.com/products/open-source/maui/

3. Neal Kobel. Distributed File Systems: Distributed Computing Architecture / Neal Kobel // CreateSpace Independent Publishing Platform, December 2016. cm.41-67
4. EC-Council. Computer Forensics: Investigating File and Operating Systems, Wireless Networks, and Storage (CHFI), 2nd Edition (Computer Hacking Forensic Investigator) 2nd Edition / EC-Council // Course Technology, April 2016. cm.18-26
5. A. Tanenbaum, Computer Networks 5th By Andrew S. Tanenbaum (International Economy Edition) / A. Tanenbaum // Prentice Hall, January 2010. cm.648-697

References:

1. Ilya Ganelin. Spark: Big Data Cluster Computing in Production / Ilya Ganelin // Wiley, March 2016. p.21-32.
2. Site Maui [Electronic resource] // Access mode: http://www.adaptivecomputing.com/products/open-source/maui/
3. Neal Kobel. Distributed File Systems: Distributed Computing Architecture / Neal Kobel // CreateSpace Independent Publishing Platform, December 2016. p.41-67
4. EC-Council. Computer Forensics: Investigating File and Operating Systems, Wireless Networks, and Storage (CHFI), 2nd Edition (Computer Hacking Forensic Investigator) 2nd Edition / EC-Council // Course Technology, April 2016. p.18-26
5. A. Tanenbaum, Computer Networks 5th By Andrew S. Tanenbaum (International Economy Edition) / A. Tanenbaum // Prentice Hall, January 2010. p.648-697