Opis rozproszonych systemów plików Google FS, Globals FS i zFS

Spis Treści

Spis Treści

Rozproszony system plików

Definicja

Jest to sieciowy system plików rozproszony na wiele fizycznych maszyn, na których przechowywana jest jedynie część wszystkich danych. Podział ten jest niewidoczny dla użytkownika korzystającego z danego systemu.

Cechy

- Przezroczystość dostępu
- Przezroczystość położenia
- Przezroczystość awarii
- Przezroczystość wydajności
- Przezroczystość współbieżności
- Skalowalność

Spis Treści

Zarys systemu GFS

Założenia

- Usterki są normą
- Pliki są DUŻE (pare GB to norma)
- Używamy częściej dopisywania do pliku niż nadpisywania
- Dwa typy odczytu: Odczyt ciągłego i długiego fragmentu lub odczyt krótkiego losowego fragmentu pliku.
- Wielu użytkowników korzysta jednocześnie z tego samego pliku.
- Dobre wykorzystanie transferu

Zarys systemu GFS

Założenia

- Usterki są normą
- Pliki są DUŻE (pare GB to norma)
- Używamy częściej dopisywania do pliku niż nadpisywania
- Dwa typy odczytu: Odczyt ciągłego i długiego fragmentu lub odczyt krótkiego losowego fragmentu pliku.
- Wielu użytkowników korzysta jednocześnie z tego samego pliku.
- Dobre wykorzystanie transferu

Zarys systemu GFS

Założenia

- Usterki są normą
- Pliki są DUŻE (pare GB to norma)
- Używamy częściej dopisywania do pliku niż nadpisywania
- Dwa typy odczytu: Odczyt ciągłego i długiego fragmentu lub odczyt krótkiego losowego fragmentu pliku.
- Wielu użytkowników korzysta jednocześnie z tego samego pliku.
- Dobre wykorzystanie transferu

Zarys systemu GFS

Założenia

- Usterki są normą
- Pliki są DUŻE (pare GB to norma)
- Używamy częściej dopisywania do pliku niż nadpisywania
- Dwa typy odczytu: Odczyt ciągłego i długiego fragmentu lub odczyt krótkiego losowego fragmentu pliku.
- Wielu użytkowników korzysta jednocześnie z tego samego pliku.
- Dobre wykorzystanie transferu

Zarys systemu GFS

Założenia

- Usterki są normą
- Pliki są DUŻE (pare GB to norma)
- Używamy częściej dopisywania do pliku niż nadpisywania
- Dwa typy odczytu: Odczyt ciągłego i długiego fragmentu lub odczyt krótkiego losowego fragmentu pliku.
- Wielu użytkowników korzysta jednocześnie z tego samego pliku.
- Dobre wykorzystanie transferu

Zarys systemu GFS

Założenia

- Usterki są normą
- Pliki są DUŻE (pare GB to norma)
- Używamy częściej dopisywania do pliku niż nadpisywania
- Dwa typy odczytu: Odczyt ciągłego i długiego fragmentu lub odczyt krótkiego losowego fragmentu pliku.
- Wielu użytkowników korzysta jednocześnie z tego samego pliku.
- Dobre wykorzystanie transferu

Klaster

Serwer główny

- Pojedynczy serwer rozwiązuje problem synchronizacji
- Pozwala na zaawansowane zarządzanie fragmentami
- Nie uczestniczy w odczytach i zapisach

Chunkserver

Wiele serwerów fragmentów (jeden fragment to 64MB!) odpowiedzialnych właściwie wyłącznie za transfer danych.

Klaster

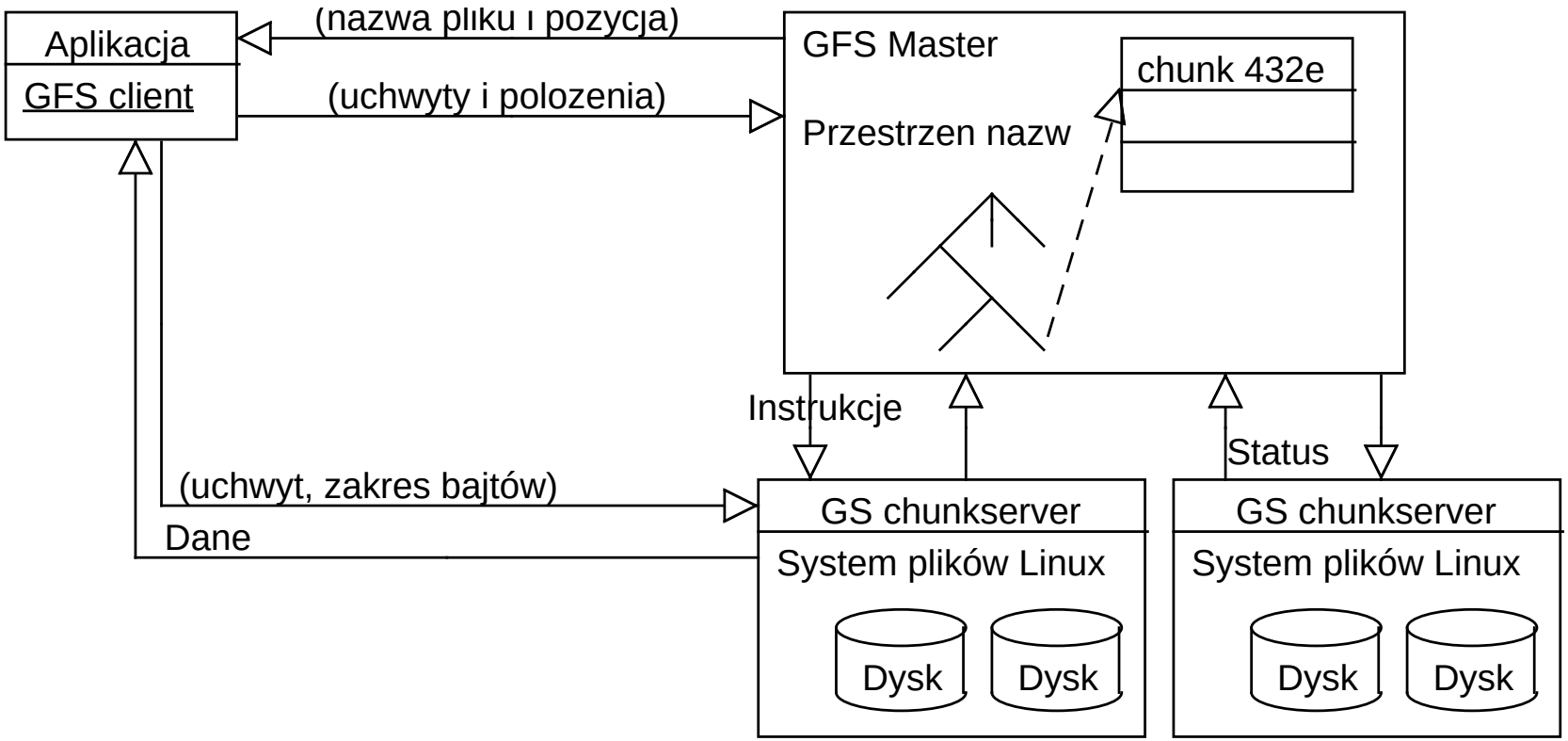
Serwer główny

- Pojedynczy serwer rozwiązuje problem synchronizacji
- Pozwala na zaawansowane zarządzanie fragmentami
- Nie uczestniczy w odczytach i zapisach

Chunkserver

Wiele serwerów fragmentów (jeden fragment to 64MB!) odpowiedzialnych właściwie wyłącznie za transfer danych.

Schemat


Aplikacja
GFS client
(nazwa pliku i pozycja)
(uchwyty i polozenia)
GFS Master
chunk 432e
Przestrzen nazw
Instrukcje
Status
(uchwyt, zakres bajtów)
Dane
GS chunkserver
System plików Linux
Dysk
Dysk
GS chunkserver
System plików Linux
Dysk
Dysk

Metadane

Metadane

Informacje o przestrzeni nazw plików, fragmentów, mapowanie z plików na fragmenty i położenie kopii każdego fragmentu.

1. Przechowywane w pamięci operacyjnej serwera głównego.
2. Kontrola nad działaniem serwerów fragmentów. Nie przechowyjemy ciągle informacji o położeniu fragmentów. Zamiast tego odpytujemy serwery fragmentów przy ich uruchamianiu.
3. Rejestr operacji - Przechowuje informacje o przestrzeni nazw oraz mapowaniu plików na fragmenty.

Metadane

Metadane

Informacje o przestrzeni nazw plików, fragmentów, mapowanie z plików na fragmenty i położenie kopii każdego fragmentu.

1. Przechowywane w pamięci operacyjnej serwera głównego.
2. Kontrola nad działaniem serwerów fragmentów. Nie przechowyjemy ciągle informacji o położeniu fragmentów. Zamiast tego odpytujemy serwery fragmentów przy ich uruchamianiu.
3. Rejestr operacji - Przechowuje informacje o przestrzeni nazw oraz mapowaniu plików na fragmenty.

Metadane

Metadane

Informacje o przestrzeni nazw plików, fragmentów, mapowanie z plików na fragmenty i położenie kopii każdego fragmentu.

1. Przechowywane w pamięci operacyjnej serwera głównego.
2. Kontrola nad działaniem serwerów fragmentów. Nie przechowyjemy ciągle informacji o położeniu fragmentów. Zamiast tego odpytujemy serwery fragmentów przy ich uruchamianiu.
3. Rejestr operacji - Przechowuje informacje o przestrzeni nazw oraz mapowaniu plików na fragmenty.

Co nam to wszystko daje?

- Operacje katalogowe (tworzenie plików, usuwanie itp.) są atomowe.
- Dobrze określony stan pliku po wykonaniu operacji, niezależnie od tego czy się ona powiodła czy nie.

	Zapis	Dopisanie	
seryjny sukces	zdefiniowany	zdefiniowany lub nierzeczywisty	
współbieżny sukces	spójny ale niezdefiniowany		
porażka	niespójny		

- Aplikacje specjalnie przystosowane, dopisują do plików, ale nie modyfikują wcześniej zapisanych danych.

Co nam to wszystko daje?

- Operacje katalogowe (tworzenie plików, usuwanie itp.) są atomowe.
- Dobrze określony stan pliku po wykonaniu operacji, niezależnie od tego czy się ona powiodła czy nie.

	Zapis	Dopisanie	
seryjny sukces	zdefiniowany	zdefiniowany lub nierzeczywisty	
współbieżny sukces	spójny ale niezdefiniowany		
porażka	niespójny		

- Aplikacje specjalnie przystosowane, dopisują do plików, ale nie modyfikują wcześniej zapisanych danych.

Co nam to wszystko daje?

- Operacje katalogowe (tworzenie plików, usuwanie itp.) są atomowe.
- Dobrze określony stan pliku po wykonaniu operacji, niezależnie od tego czy się ona powiodła czy nie.

	Zapis	Dopisanie	
seryjny sukces	zdefiniowany	zdefiniowany lub nierzeczywisty	
współbieżny sukces	spójny ale niezdefiniowany		
porażka	niespójny		

- Aplikacje specjalnie przystosowane, dopisują do plików, ale nie modyfikują wcześniej zapisanych danych.

Co nam to wszystko daje?

- Operacje katalogowe (tworzenie plików, usuwanie itp.) są atomowe.
- Dobrze określony stan pliku po wykonaniu operacji, niezależnie od tego czy się ona powiodła czy nie.

	Zapis	Dopisanie	
seryjny sukces	zdefiniowany	zdefiniowany lub nierzeczywisty	
współbieżny sukces	spójny ale niezdefiniowany		
porażka	niespójny		

- Aplikacje specjalnie przystosowane, dopisują do plików, ale nie modyfikują wcześniej zapisanych danych.

Działanie

- Blokady na modyfikowane pliki.
- Efektywne wykorzystanie sieci.
- GFS gwarantuje, że dane są zapisane conajmniej raz.
- Zachowywanie wersji pliku.
- Serwer główny zarządza przestrzenią nazw.
- Leniwe odzyskiwanie przestrzeni.
- Wiele kopii jednego fragmentu. Domyślnie 3 kopie każdego fragmentu.

Działanie

- Blokady na modyfikowane pliki.
- Efektywne wykorzystanie sieci.
- GFS gwarantuje, że dane są zapisane conajmniej raz.
- Zachowywanie wersji pliku.
- Serwer główny zarządza przestrzenią nazw.
- Leniwe odzyskiwanie przestrzeni.
- Wiele kopii jednego fragmentu. Domyślnie 3 kopie każdego fragmentu.

Działanie

- Blokady na modyfikowane pliki.
- Efektywne wykorzystanie sieci.
- GFS gwarantuje, że dane są zapisane conajmniej raz.
- Zachowywanie wersji pliku.
- Serwer główny zarządza przestrzenią nazw.
- Leniwe odzyskiwanie przestrzeni.
- Wiele kopii jednego fragmentu. Domyślnie 3 kopie każdego fragmentu.

Działanie

- Blokady na modyfikowane pliki.
- Efektywne wykorzystanie sieci.
- GFS gwarantuje, że dane są zapisane conajmniej raz.
- Zachowywanie wersji pliku.
- Serwer główny zarządza przestrzenią nazw.
- Leniwe odzyskiwanie przestrzeni.
- Wiele kopii jednego fragmentu. Domyślnie 3 kopie każdego fragmentu.

Działanie

- Blokady na modyfikowane pliki.
- Efektywne wykorzystanie sieci.
- GFS gwarantuje, że dane są zapisane conajmniej raz.
- Zachowywanie wersji pliku.
- Serwer główny zarządza przestrzenią nazw.
- Leniwe odzyskiwanie przestrzeni.
- Wiele kopii jednego fragmentu. Domyślnie 3 kopie każdego fragmentu.

Działanie

- Blokady na modyfikowane pliki.
- Efektywne wykorzystanie sieci.
- GFS gwarantuje, że dane są zapisane conajmniej raz.
- Zachowywanie wersji pliku.
- Serwer główny zarządza przestrzenią nazw.
- Leniwe odzyskiwanie przestrzeni.
- Wiele kopii jednego fragmentu. Domyślnie 3 kopie każdego fragmentu.

Działanie

- Blokady na modyfikowane pliki.
- Efektywne wykorzystanie sieci.
- GFS gwarantuje, że dane są zapisane conajmniej raz.
- Zachowywanie wersji pliku.
- Serwer główny zarządza przestrzenią nazw.
- Leniwe odzyskiwanie przestrzeni.
- Wiele kopii jednego fragmentu. Domyślnie 3 kopie każdego fragmentu.

Stabilność

- Szybki restart systemu (czas rzędu sekund).
- Wiele kopii jednego fragmentu. W razie uszkodzenia, nowe kopie są tworzone jak najszybciej.
- Duchy serwera głównego.
- Sumy kontrolne na serwerach fragmentów.

Sumy kontrolne

Dla każdych 64kB tworzona jest suma kontrolna.

- Logowanie informacji diagnostycznych.

Stabilność

- Szybki restart systemu (czas rzędu sekund).
- Wiele kopii jednego fragmentu. W razie uszkodzenia, nowe kopie są tworzone jak najszybciej.
- Duchy serwera głównego.
- Sumy kontrolne na serwerach fragmentów.

Sumy kontrolne

Dla każdych 64kB tworzona jest suma kontrolna.

- Logowanie informacji diagnostycznych.

Stabilność

- Szybki restart systemu (czas rzędu sekund).
- Wiele kopii jednego fragmentu. W razie uszkodzenia, nowe kopie są tworzone jak najszybciej.
- Duchy serwera głównego.
- Sumy kontrolne na serwerach fragmentów.

Sumy kontrolne

Dla każdych 64kB tworzona jest suma kontrolna.

- Logowanie informacji diagnostycznych.

Stabilność

- Szybki restart systemu (czas rzędu sekund).
- Wiele kopii jednego fragmentu. W razie uszkodzenia, nowe kopie są tworzone jak najszybciej.
- Duchy serwera głównego.
- Sumy kontrolne na serwerach fragmentów.

Sumy kontrolne

Dla każdych 64kB tworzona jest suma kontrolna.

- Logowanie informacji diagnostycznych.

Stabilność

- Szybki restart systemu (czas rzędu sekund).
- Wiele kopii jednego fragmentu. W razie uszkodzenia, nowe kopie są tworzone jak najszybciej.
- Duchy serwera głównego.
- Sumy kontrolne na serwerach fragmentów.

Sumy kontrolne

Dla każdych 64kB tworzona jest suma kontrolna.

- Logowanie informacji diagnostycznych.

Podsumowanie

- System daje dużą przepustowość dla wielu użytkowników korzystających z niego jednocześnie.
- Zastosowanie pojedynczego serwera głównego, który mimo to nie jest wąskim gardłem.
- System jest używany zarówno w badaniach, rozwoju jak i w produkcji.

Podsumowanie

- System daje dużą przepustowość dla wielu użytkowników korzystających z niego jednocześnie.
- Zastosowanie pojedynczego serwera głównego, który mimo to nie jest wąskim gardłem.
- System jest używany zarówno w badaniach, rozwoju jak i w produkcji.

Podsumowanie

- System daje dużą przepustowość dla wielu użytkowników korzystających z niego jednocześnie.
- Zastosowanie pojedynczego serwera głównego, który mimo to nie jest wąskim gardłem.
- System jest używany zarówno w badaniach, rozwoju jak i w produkcji.

Spis Treści

Historia

- 1995 - profesor Matthew O'Keefe wraz z grupą studentów rozpoczyna prace nad Global File System (GFS)
- 1996 - Cluster File System i Fibre Channel Storage
- 1997 - założenie przez profesora Matthew O'Keefe organizacji Sistina której własnością stał się GFS
- 2001 - zmiana licencji, GFS staje się produktem komercyjnym
- 2001 - powstanie projektu OpenGFS
- grudzień 2003 - Red Hat kapuje Sistine (GFS oraz LVM)
- styczeń 2004 - powrót do licencji GPL
- 2006 włącznie GFS w wersji GFS2 do jądra 2.6.16

Historia

- 1995 - profesor Matthew O'Keefe wraz z grupą studentów rozpoczyna prace nad Global File System (GFS)
- 1996 - Cluster File System i Fibre Channel Storage
- 1997 - założenie przez profesora Matthew O'Keefe organizacji Sistina której własnością stał się GFS
- 2001 - zmiana licencji, GFS staje się produktem komercyjnym
- 2001 - powstanie projektu OpenGFS
- grudzień 2003 - Red Hat kapuje Sistine (GFS oraz LVM)
- styczeń 2004 - powrót do licencji GPL
- 2006 włącznie GFS w wersji GFS2 do jądra 2.6.16

Historia

- 1995 - profesor Matthew O'Keefe wraz z grupą studentów rozpoczyna prace nad Global File System (GFS)
- 1996 - Cluster File System i Fibre Channel Storage
- 1997 - założenie przez profesora Matthew O'Keefe organizacji Sistina której własnością stał się GFS
- 2001 - zmiana licencji, GFS staje się produktem komercyjnym
- 2001 - powstanie projektu OpenGFS
- grudzień 2003 - Red Hat kapuje Sistine (GFS oraz LVM)
- styczeń 2004 - powrót do licencji GPL
- 2006 włącznie GFS w wersji GFS2 do jądra 2.6.16

Historia wersji

- v1.0 (1996) SGI IRIX only
- v3.0 Linux-port
- v4 Journaling
- v5 Redundant Lock Manager
- v6.1 (2005) Distributed Lock Manager

Historia wersji

- v1.0 (1996) SGI IRIX only
- v3.0 Linux-port
- v4 Journaling
- v5 Redundant Lock Manager
- v6.1 (2005) Distributed Lock Manager

Historia wersji

- v1.0 (1996) SGI IRIX only
- v3.0 Linux-port
- v4 Journaling
- v5 Redundant Lock Manager
- v6.1 (2005) Distributed Lock Manager

GFS - Global file system

GFS - Global file system

GFS - jest to shared disk file systems przeznaczony dla klastrów pracujących pod systemem Linux. GFS wyposażony jest we własny mechanizm DLM (Distributed Locking Manager).

shared disk file system

distributed file systems - poszczególne węzły mają bezpośredni dostęp tylko do części całego systemu plików
shared disk file systems - wszystkie węzły mają równoległy bezpośredni dostęp do całego systemu plików

GFS - Global file system

GFS - Global file system

GFS - jest to shared disk file systems przeznaczony dla klastrów pracujących pod systemem Linux. GFS wyposażony jest we własny mechanizm DLM (Distributed Locking Manager).

shared disk file system

distributed file systems - poszczególne węzły mają bezpośredni dostęp tylko do części całego systemu plików
shared disk file systems - wszystkie węzły mają równoległy bezpośredni dostęp do całego systemu plików

Storage area network

Storage area network

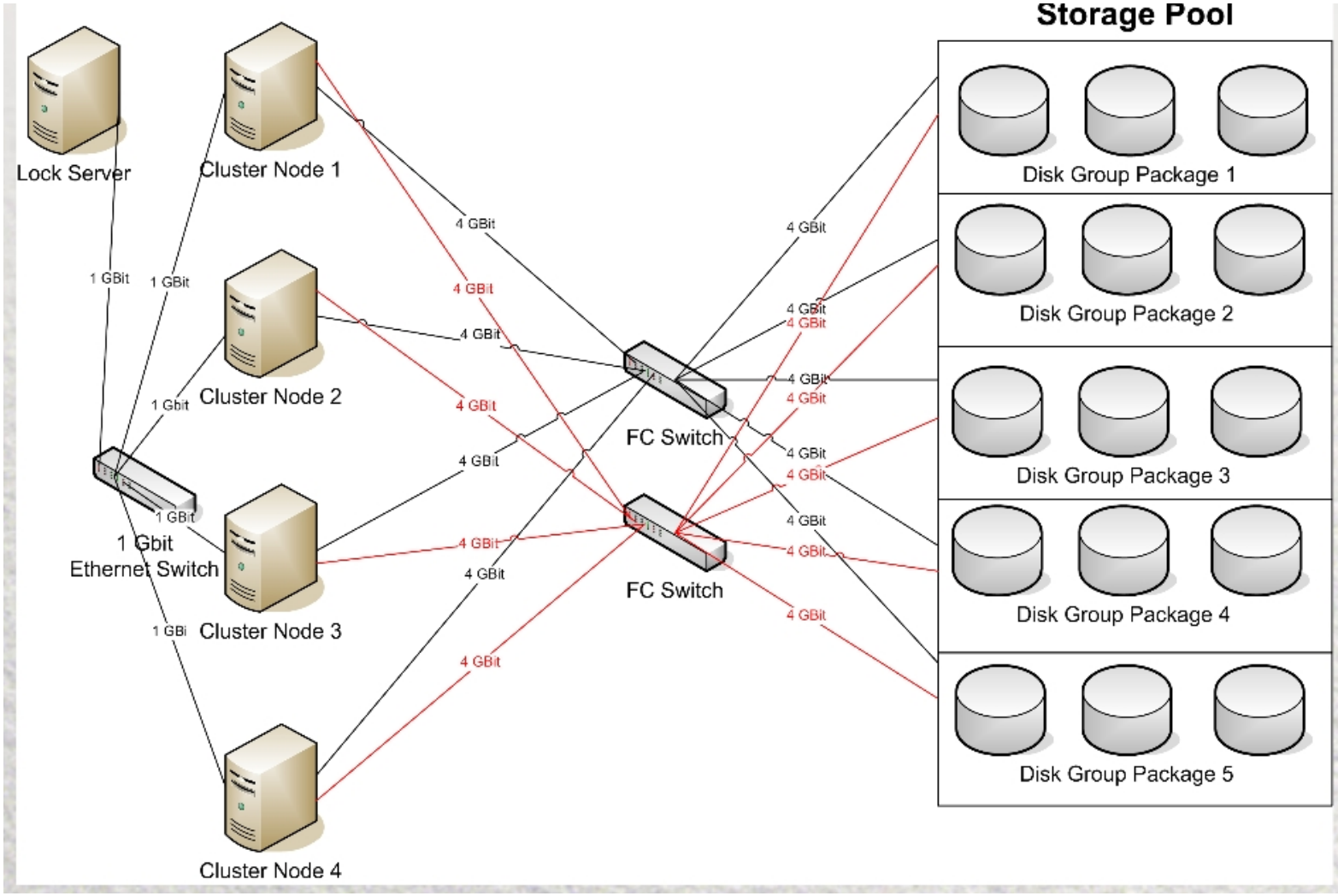
SAN - architektura łącząca macierze dyskowe (lub zbiory taśm itp) do serwera w taki sposób by system operacyjny widział je jako jedną całość. Najczęściej do łącznia serwerów i urządzeń używany jest protokół SCSI obudowany przez warstwę umożliwiająca wykorzystanie go w sieciach:

- Fibre Channel Protocol (FCP), mapping SCSI over Fibre Channel, znany również jako "iFCP" lub "SANoIP"
- iSCSI, mapping SCSI over TCP/IP
- HyperSCSI, mapping SCSI over Ethernet

Inne technologie to np:

- FICON mapping over Fibre Channel
- ATA over Ethernet, mapping ATA over Ethernet
- SCSI and/or TCP/IP mapping over InfiniBand (IB)

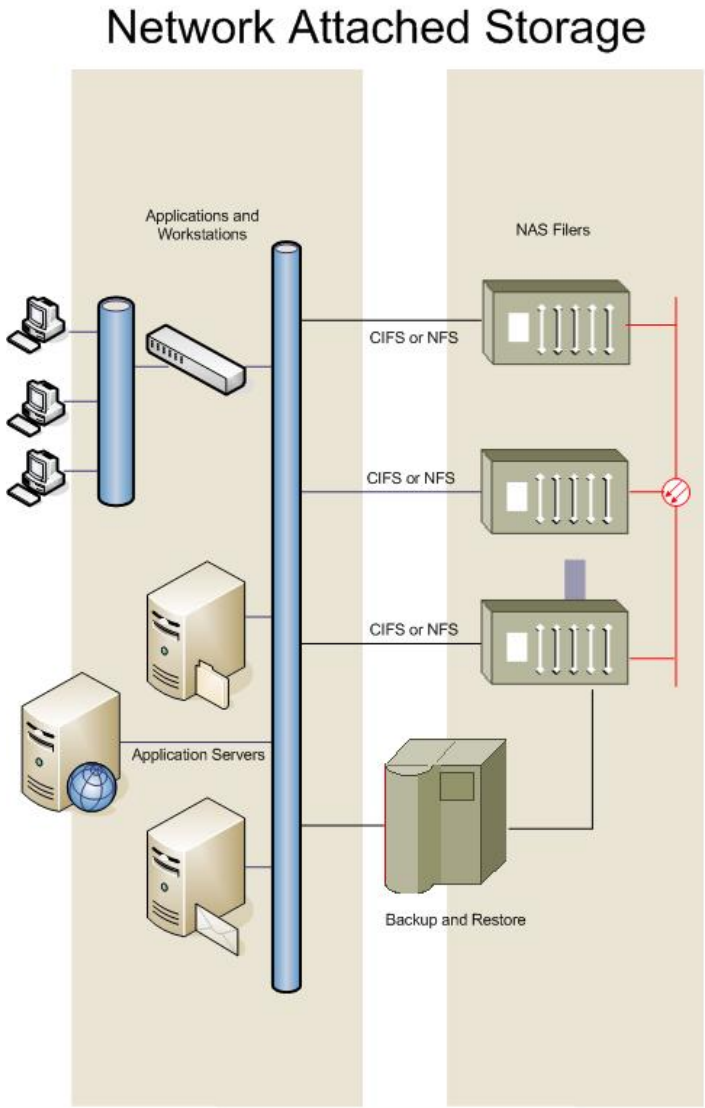
GFS - Fibre Channel (FC)



network-attached storage

network-attached storage

NAS - jasne jest, że zasoby są zdalne, wykorzystuje się protokoły zorientowane plikowo takie jak NFS lub SMB/CIFS. Operacje odbywają się na plikach, a nie blokach danych.



Najważniejsze cechy

- wolne oprogramowanie, na licencji GPL
- 64bit, big-endian
- zaprojektowany z myślą o SAN
- często używany wraz z Fibre Channel, iSCSI, or AoE
- może składać się nawet z 300 serwerów, nadal zapewniając wysoką wydajność każdemu węzłowi
- obsługuje rozszerzony zestaw atrybutów oraz POSIX ACLs, odpowiedni dla selinux
- zapewnia spójność danych, odczyt ostatnio zapisanych danych

Najważniejsze cechy

- wolne oprogramowanie, na licencji GPL
- 64bit, big-endian
- zaprojektowany z myślą o SAN
- często używany wraz z Fibre Channel, iSCSI, or AoE
- może składać się nawet z 300 serwerów, nadal zapewniając wysoką wydajność każdemu węzłowi
- obsługuje rozszerzony zestaw atrybutów oraz POSIX ACLs, odpowiedni dla selinux
- zapewnia spójność danych, odczyt ostatnio zapisanych danych

Najważniejsze cechy

- wolne oprogramowanie, na licencji GPL
- 64bit, big-endian
- zaprojektowany z myślą o SAN
- często używany wraz z Fibre Channel, iSCSI, or AoE
- może składać się nawet z 300 serwerów, nadal zapewniając wysoką wydajność każdemu węzłowi
- obsługuje rozszerzony zestaw atrybutów oraz POSIX ACLs, odpowiedni dla selinux
- zapewnia spójność danych, odczyt ostatnio zapisanych danych

Najważniejsze cechy

- wolne oprogramowanie, na licencji GPL
- 64bit, big-endian
- zaprojektowany z myślą o SAN
- często używany wraz z Fibre Channel, iSCSI, or AoE
- może składać się nawet z 300 serwerów, nadal zapewniając wysoką wydajność każdemu węzłowi
- obsługuje rozszerzony zestaw atrybutów oraz POSIX ACLs, odpowiedni dla selinux
- zapewnia spójność danych, odczyt ostatnio zapisanych danych

Najważniejsze cechy

- wolne oprogramowanie, na licencji GPL
- 64bit, big-endian
- zaprojektowany z myślą o SAN
- często używany wraz z Fibre Channel, iSCSI, or AoE
- może składać się nawet z 300 serwerów, nadal zapewniając wysoką wydajność każdemu węzłowi
- obsługuje rozszerzony zestaw atrybutów oraz POSIX ACLs, odpowiedni dla selinux
- zapewnia spójność danych, odczyt ostatnio zapisanych danych

Najważniejsze cechy

- wolne oprogramowanie, na licencji GPL
- 64bit, big-endian
- zaprojektowany z myślą o SAN
- często używany wraz z Fibre Channel, iSCSI, or AoE
- może składać się nawet z 300 serwerów, nadal zapewniając wysoką wydajność każdemu węzłowi
- obsługuje rozszerzony zestaw atrybutów oraz POSIX ACLs, odpowiedni dla selinux
- zapewnia spójność danych, odczyt ostatnio zapisanych danych

Najważniejsze cechy

- brak podziału ról na server/klient, węzły są identyczne
- każdy węzeł ma równy dostęp do całego zbioru
- struktury metadanych są podzielone pomiędzy poszczególne dyski
- użycie w klastrach wymaga **lock manager**'a
  - GULM - scentralizowany, już nie rozwijany
  - Distributed Lock Manager (DLM) - najczęściej używany
  - "nolock" - użycie GFS jako zwykłego systemu plików

Najważniejsze cechy

- brak podziału ról na server/klient, węzły są identyczne
- każdy węzeł ma równy dostęp do całego zbioru
- struktury metadanych są podzielone pomiędzy poszczególne dyski
- użycie w klastrach wymaga **lock manager**'a
  - GULM - scentralizowany, już nie rozwijany
  - Distributed Lock Manager (DLM) - najczęściej używany
  - "nolock" - użycie GFS jako zwykłego systemu plików

Najważniejsze cechy

- brak podziału ról na server/klient, węzły są identyczne
- każdy węzeł ma równy dostęp do całego zbioru
- struktury metadanych są podzielone pomiędzy poszczególne dyski
- użycie w klastrach wymaga **lock manager**'a
  - GULM - scentralizowany, już nie rozwijany
  - Distributed Lock Manager (DLM) - najczęściej używany
  - "nolock" - użycie GFS jako zwykłego systemu plików

distributed lock manager

distributed lock manager

DLM - synchronizuje dostęp do dzielonych zasobów, jest odpowiedzialny nie tylko za blokowanie, ale rownież zarządzanie dostępem do dysków. Wykorzystuje różne typy blokad np dzielony odczyt, wyłączność na pisanie.

Example

Przykład: Zanim węzeł zacznie pisać do plików w systemie GFS musi uzyskać (write lock) wyłączność na pisanie do tego pliku. Jeśli inny węzeł (węzeł 2) w tym czasie czyta lub pisze do tego samego pliku, manager blokowania informuje węzeł 2 by zwolnił blokadę na plik. Gdy ten to zrobi manager, przekazuje blokadę węzłowi pierwszemu.

distributed lock manager

distributed lock manager

DLM - synchronizuje dostęp do dzielonych zasobów, jest odpowiedzialny nie tylko za blokowanie, ale rownież zarządzanie dostępem do dysków. Wykorzystuje różne typy blokad np dzielony odczyt, wyłączność na pisanie.

Example

Przykład: Zanim węzeł zacznie pisać do plików w systemie GFS musi uzyskać (write lock) wyłączność na pisanie do tego pliku. Jeśli inny węzeł (węzeł 2) w tym czasie czyta lub pisze do tego samego pliku, manager blokowania informuje węzeł 2 by zwolnił blokadę na plik. Gdy ten to zrobi manager, przekazuje blokadę węzłowi pierwszemu.

Awarie

Infrastruktura monitorów węzłów sprawdza czy każdy funkcjonuje poprawnie, jeśli jakiś węzeł lub jego połączenie sieciowe zawodzi, co prowadzi do odłączenia węzła od klastra, członkowie warstwy wykrywają to i inicjują odgradzanie węzła, izolują węzeł odłączając go od zbioru i resetują go.

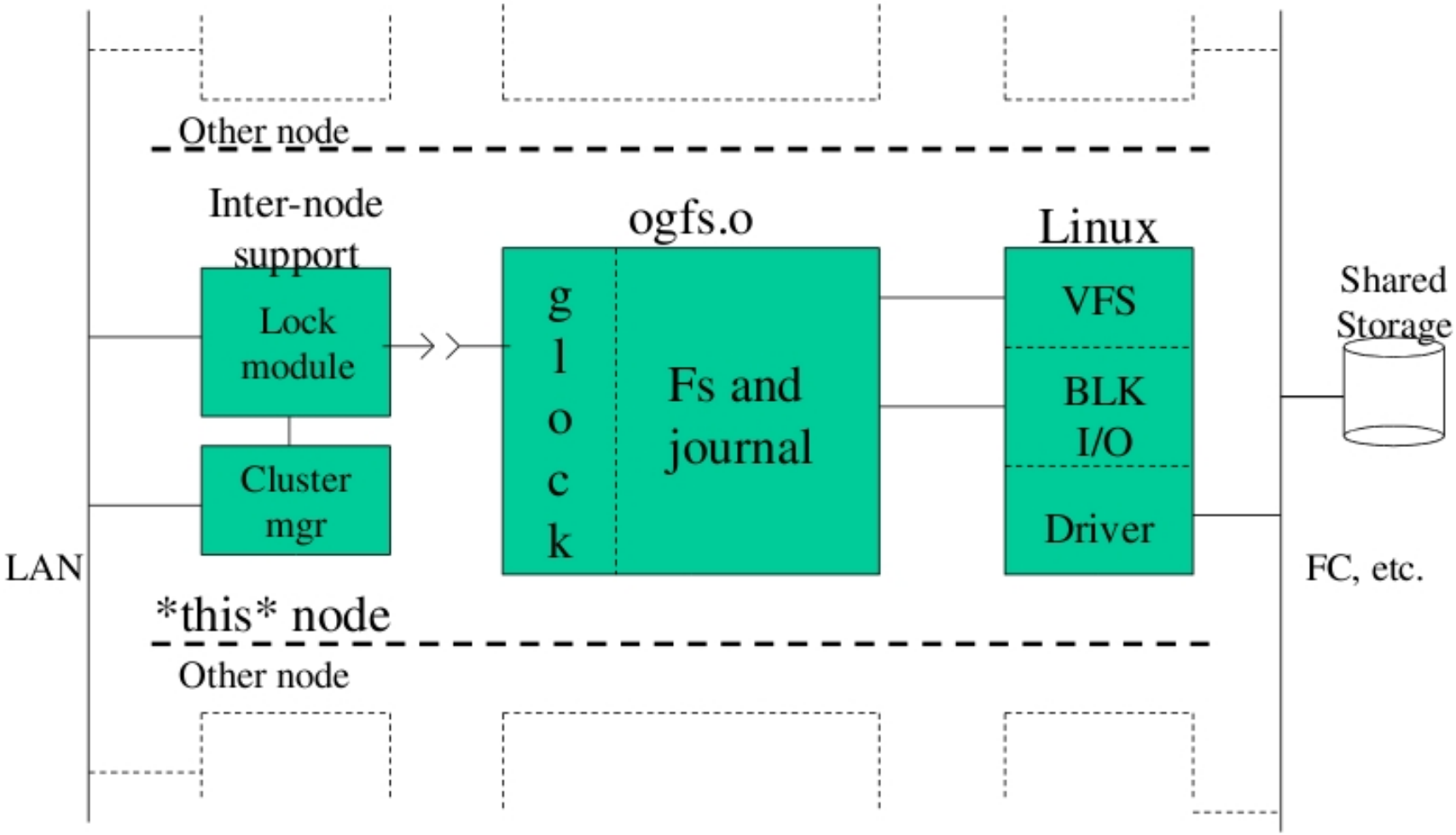
Po sprawdzeniu spójności z dziennikiem i odzyskaniu stanu blokad, odgrodzony węzeł jest z powrotem przyłączany do zbioru.

Odłączanie węzła i operacje naprawcze mogą zająć dziesiątki sekund.

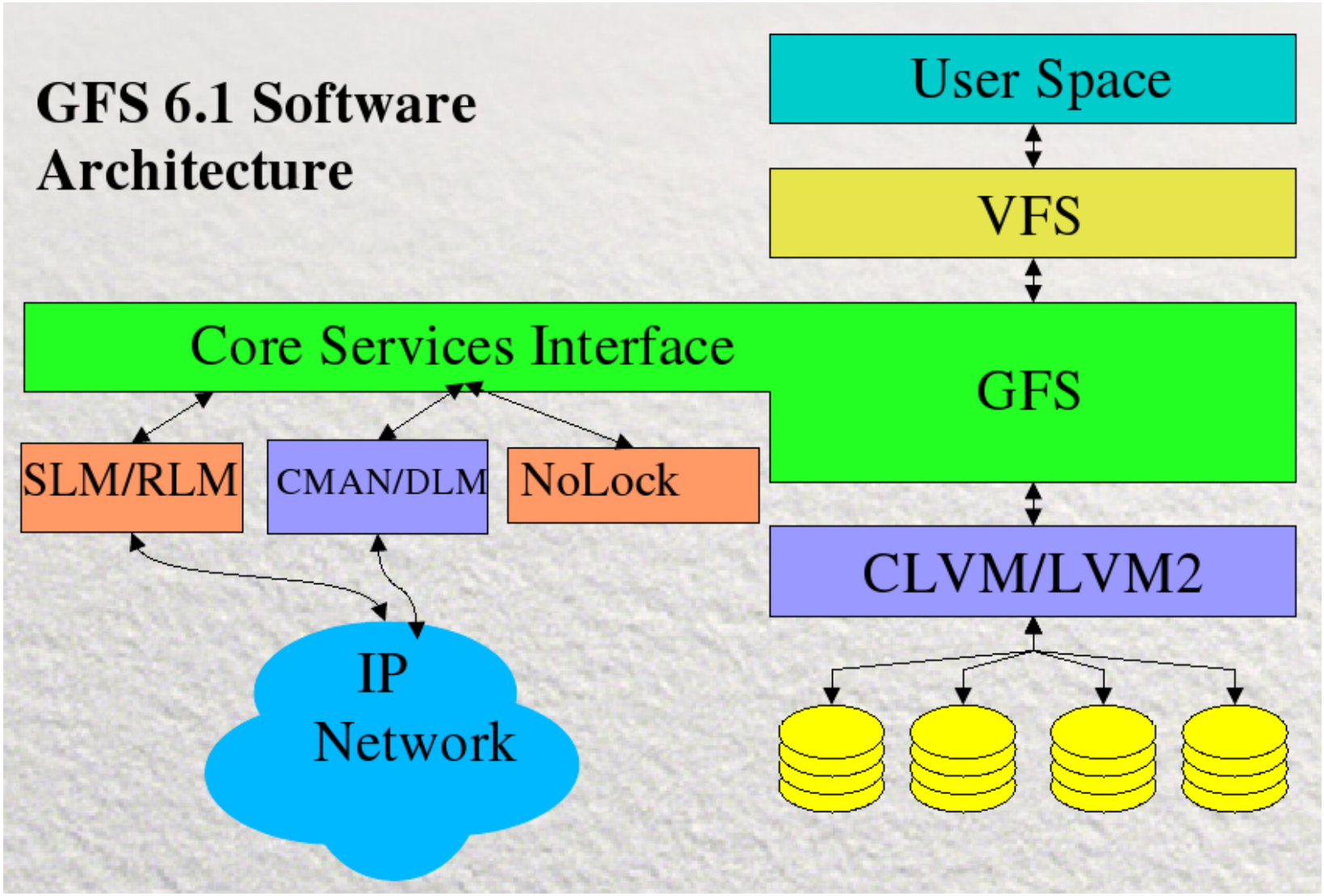
Awarie

Infrastruktura monitorów węzłów sprawdza czy każdy funkcjonuje poprawnie, jeśli jakiś węzeł lub jego połączenie sieciowe zawodzi, co prowadzi do odłączenia węzła od klastra, członkowie warstwy wykrywają to i inicjują odgradzanie węzła, izolują węzeł odłączając go od zbioru i resetują go.
Po sprawdzeniu spójności z dziennikiem i odzyskaniu stanu blokad, odgrodzony węzeł jest z powrotem przyłączany do zbioru.
Odłączanie węzła i operacje naprawcze mogą zająć dziesiątki sekund.

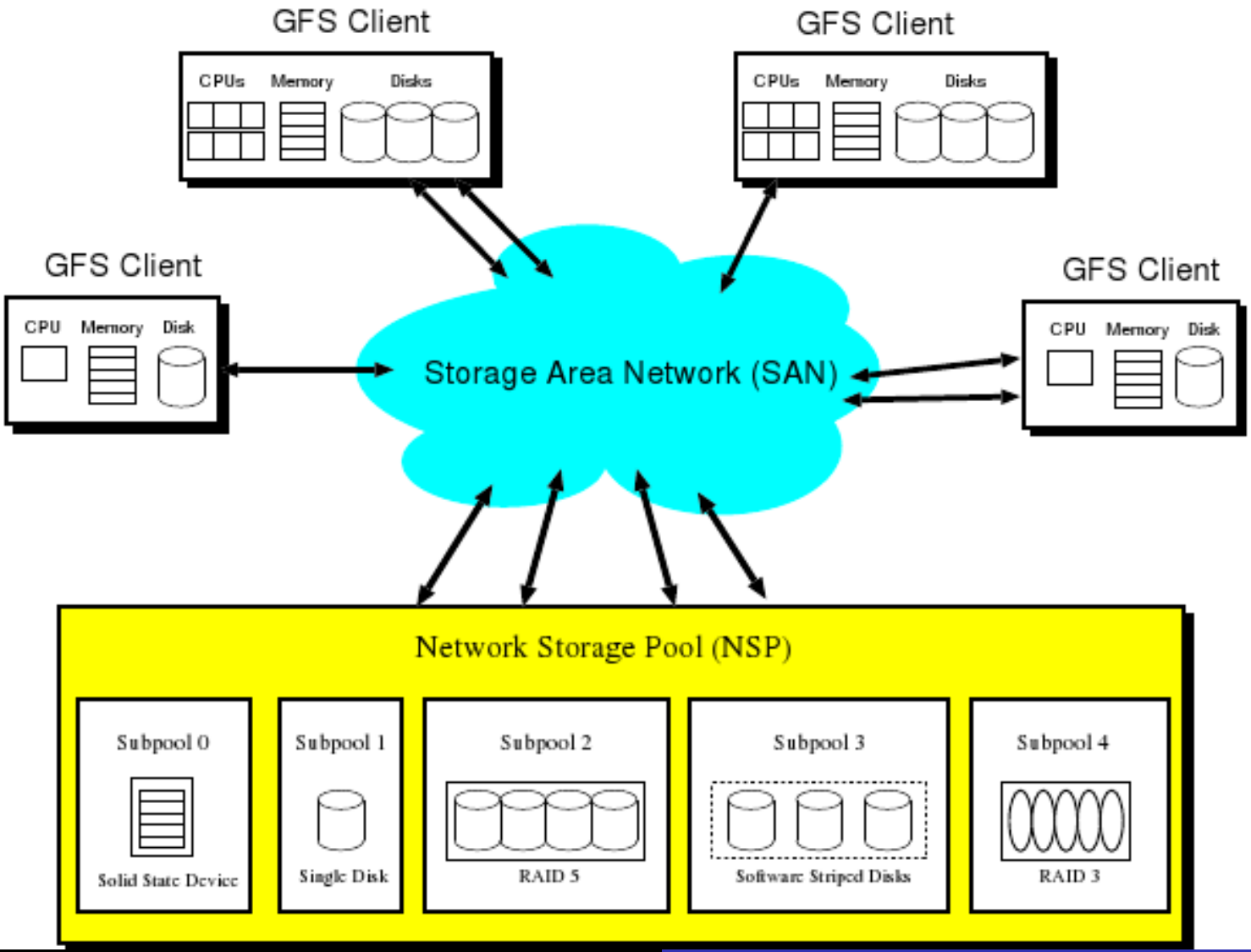
OpenGFS


Other node
Inter-node support
Lock module
Cluster mgr
ogfs.o
g l o c k
Fs and journal
Linux
VFS
BLK I/O
Driver
Shared Storage
LAN
FC, etc.
this node
Other node

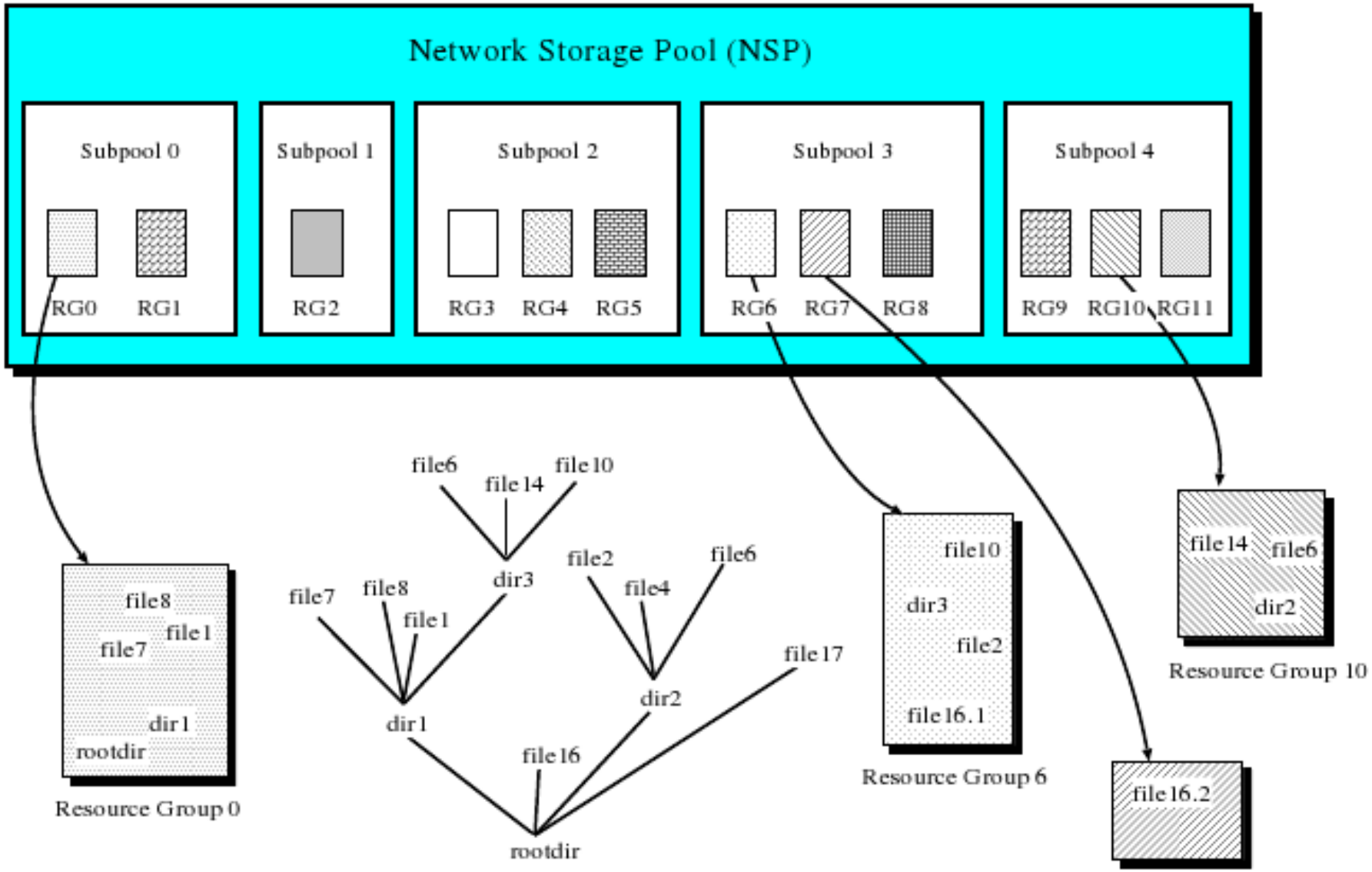
GFS 6.1


GFS 6.1 Software Architecture
User Space
VFS
Core Services Interface
GFS
SLM/RLM
CMAN/DLM
NoLock
CLVM/LVM2
IP Network

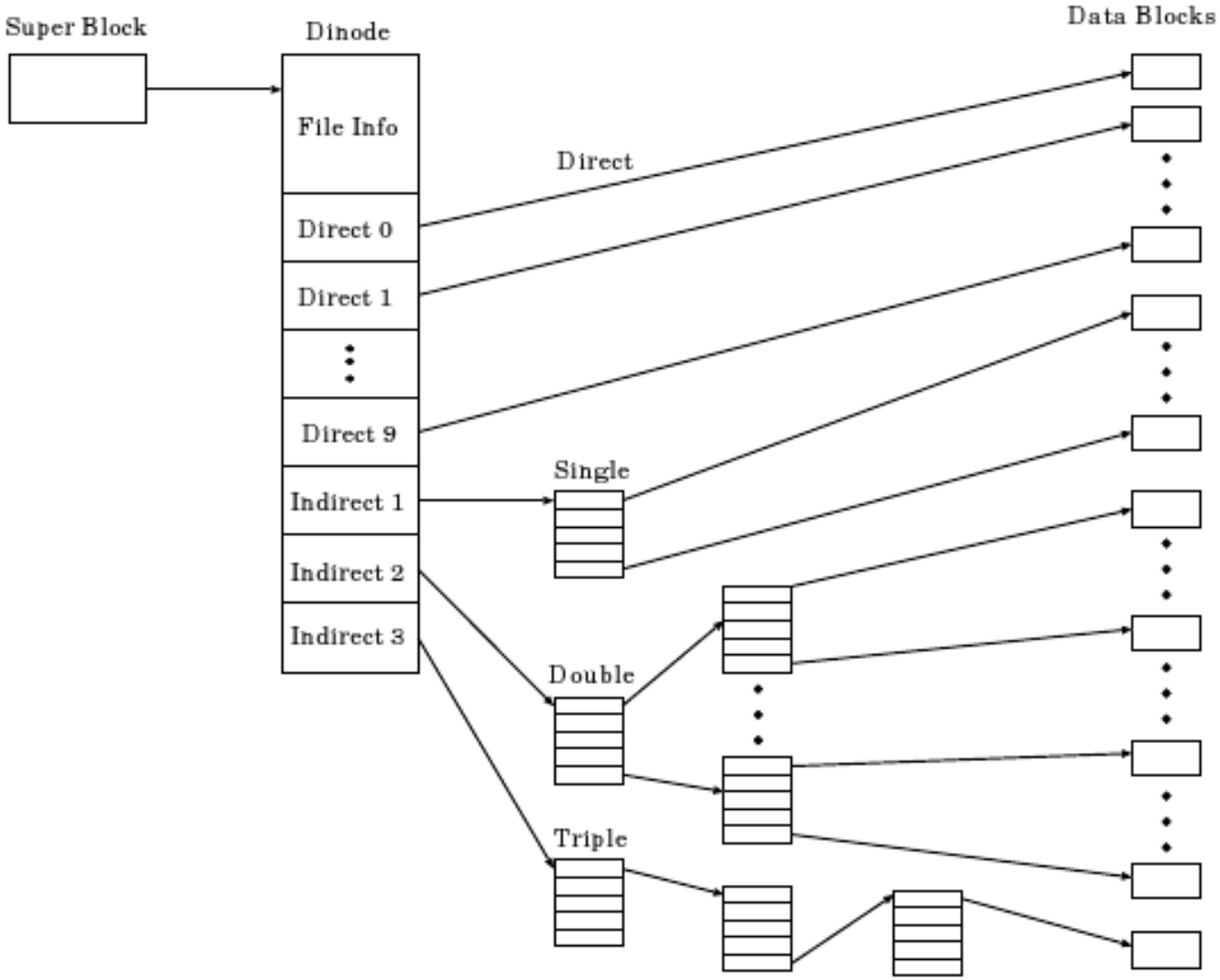
GFS Distributed System



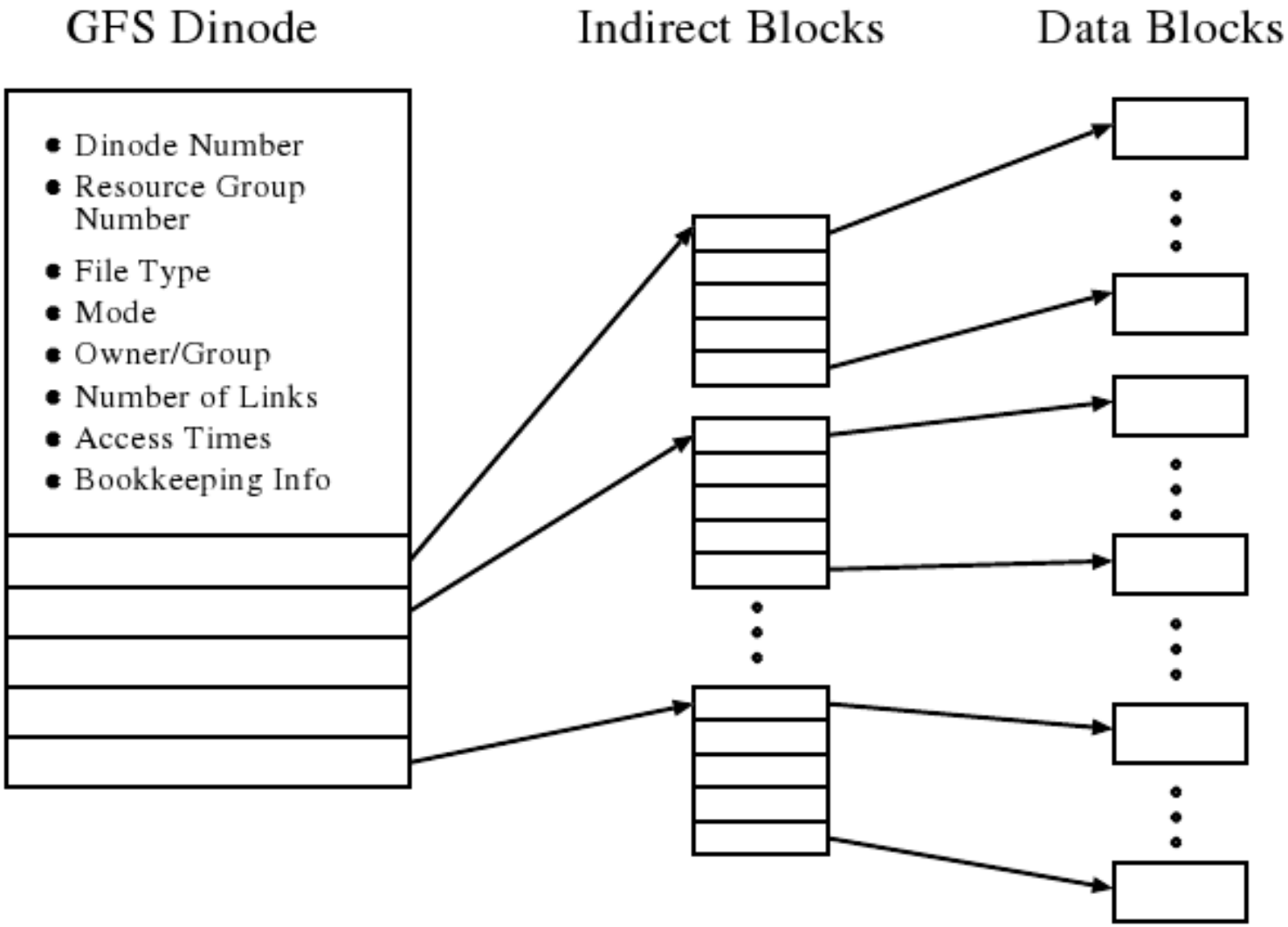
Network Storage Pool



UFS Inode and Metadata Tree



GFS Dinode and Metadata Tree



Tworzenie systemu plików

gfs_mkfs

Tworzenie sytemu plików GFS wymaga użycia programu gfs_mkfs do zainicjowania metadanych dzielonego woluminu.

gfs_mkfs -p lock_dlm -t alpha:gfs1 -j 8 /dev/vg01/lvol0

W tym przykładzie gfs_mkfs został użyty do stworzenia sytemu plików o nazwie gfs1 w klastrze o nazwie alpha który używa rozproszonego managera blokowania (DLM). System plików ma 8 dzienników co pozwala na obsługę do 8 węzłów i znajduje się na woluminie LVM2 o nazwie dev/vg01/lvol0.

Montowanie systemu plików

mount

mount -t gfs /dev/vg01/lvol0 /gfs1

Gdy system plików został utworzony, woluminy aktywowane oraz uruchomione zostały systemy klasteringu i blokowania może zostać zamontowany i udostępniony.

Rozszerzanie systemu plików

mount

gfs_grow /gfs1

System plików może być rozszerzony nawet gdy został już zamontowany i jest w użycie bez przerywania jego pracy. Komendę gfs_grow wystarczy uruchomić na jednym z węzłów, pozostałe automatycznie zauważa zmianę i zaczną używać nowych zasobów.

Spis Treści

zFS – co to znaczy?

- zFS – A Scalable distributed File System using Object Stores
- projekt badawczy firmy IBM
- bazuje na wcześniejszym projekcie: DSF – Data Sharing Facility

zFS – co to takiego?

- rozproszony system plików – brak jakiegokolwiek centralnego serwera
- wszelkie czynności związane z przechowywaniem i zarządzaniem plikami rozproszone po wszystkich współpracujących maszynach
- działać ma na istniejących systemach operacyjnych (Linux)
- sprzęt – komputery klasy PC połączone szybką siecią (m. in.)

Cele projektowe

- Skalowalność – od kilku do kilku tysięcy komputerów; do dziesiątek tysięcy użytkowników.
- Zbudowany na bazie Object Stores.
- Zwiększenie wydajności poprzez użycie pamięci wszystkich komputerów w ramach jednego globalnego cache'u.
- Dodawanie maszyn ma liniowo zwiększać wydajność.

Komponenty

Implementacja zFS dzieli się na sześć współpracujących ze sobą komponentów.

- Object Store (ObS, Object Disk, Object Store Device, OSD)
- Front End (FE)
- Lease Manager (LMGR)
- File Manager
- Cooperative Cache
- Transaction Server

Object Store (ObS)

- urządzenie na którym fizycznie przechowywane są pliki i katalogi – obiekty
- zamiennik dla tradycyjnych urządzeń blokowych
- zbiór obiektów
- obiekty = ciągi bajtów + metadane (atrybuty)

Object Store (ObS)

Interfejs

- udostępnia wysokopoziomowy interfejs z operacjami:
  - tworzenie/usuwanie obiektów
  - pisanie/czytanie danych z dowolnego miejsca w obiekcie
- protokół T10; komunikacja – komendy SCSI

Bezpieczeństwo

- każdy obiekt ma przypisane prawa dostępu
- wszystkie operacje wymagają autoryzacji

Alokacja

Alokacja przestrzeni dyskowej odbywa się na poziomie Object Store, a nie na poziomie software – system plików nie musi troszczyć się o niskopoziomowe szczegóły.

Object Store (ObS)

Jest dostępny sterownik pod Linuxa (OSD Initiator) oraz symulator Object Store, który dane przechowuje na dysku komputera na zewnątrz udostępniając interfejs ObS (OSD Simulator for Linux).

Front End (FE)

- sterownik (moduł jądra) uruchamiany na każdym komputerze klienta
- udostępnia POSIX'owe API do systemu zFS

Lease Manager (LMGR)

lease

- blokada typu *lease* – blokada (lock) o ustalonym z góry terminie ważności
- używana w systemach rozproszonych aby zapobiec sytuacji w której ktoś zakłada blokadę, ulega awarii i tej blokady nie zdejmuje

- każdy ObS ma jednego LMGR, który zajmuje się zarządzaniem blokad; ObS pamięta jego adres (w sieci)
- jedna blokada na cały ObS
- wyłączne blokady na poszczególne obiekty

Lease Manager (LMGR)

Jak wygląda dostęp do obiektów wewnątrz ObS?

- zpytaj się ObS o adres sieciowy jego LMGR
- jeśli nie istnieje stwórz lokalnie nowy LMGR
- poproś LMGR o dostęp do odpowiedniego obiektu

File Manager (FMGR)

- każdy otwarty plik ma przypisany jednego FMGR – tworzony gdy plik otwierany jest przez pierwszego klienta
- cały czas gdy plik jest otwarty założona jest na nim blokada wyłączna pobrana LMGR; w posiadaniu tej blokady jest FMGR
- FMGR zajmuje się zarządzaniem dostępami do pliku – przydziela blokady (typu *lease*) na odpowiednie części pliku

Cooperative Cache

- kluczowa własność poprawiająca skalowalność
- założenie: szybka sieć – przesyłanie danych przez sieć szybsze niż czytanie dysku
- jeśli komputer A chce czytać pewien plik, który znajduje się już w pamięci innego komputera B w sieci, to FMGR wykrywa taką sytuację i kopiuje dane do cache'a maszyny A

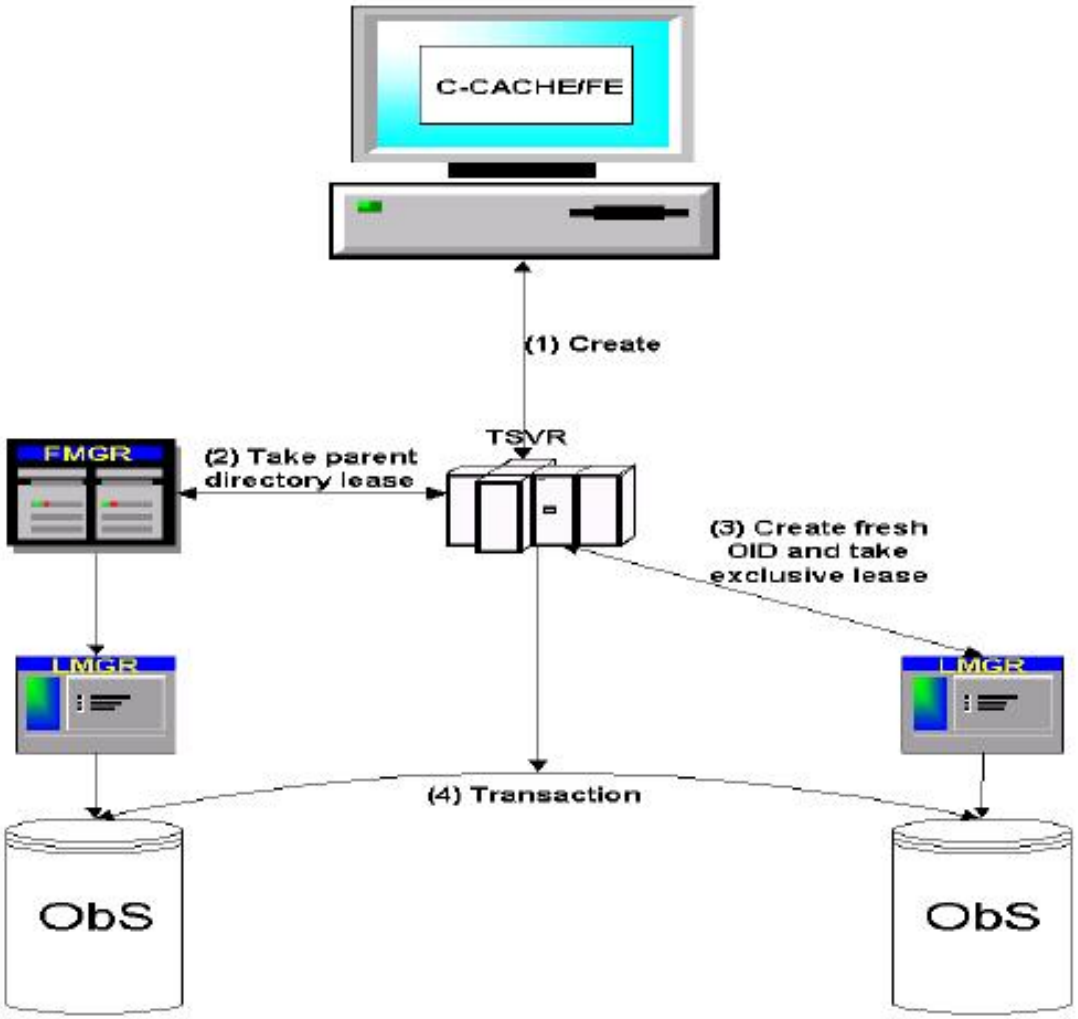
Transaction server (TSVR)

Operacje na katalogach implementowane są jako rozproszone transakcje – *transaction server* zajmuje się zarządzaniem tych transakcji, tak aby nie doszło do popsucia systemu plików w przypadku niepowodzenia którejś ze składowych operacji.

Przykład – zmiana nazwy pliku (UNIX'owe rename))

- wymaga przynajmniej:
  - założenie blokad na katalog źródłowy, docelowy oraz sam plik
  - utworzenie nowego wpisu w katalogu docelowym
  - usunięcie wpisu w katalogu źródłowym
  - zwolnienie blokad
- każda z tych operacji wykonywana jest niezależnie i każda z nich może się nie powieść
- coś musi tymi niezależnymi operacjami zarządzać – Transaction Manager

Przykład – tworzenie pliku



Przykład – tworzenie pliku

1. FE dostaje od klienta polecenie utworzenia pliku pod podaną ścieżką. Najpierw odszukuje on tę ścieżkę, w przypadku brakujących elementów konsultując się z TSVR. Otrzymuje w ten sposób katalog gdzie ma być utworzony plik.
2. FE prosi TSVR o utworzenie tego pliku.
3. TSVR robi co następuje:
   1. zakłada blokady
   2. wybiera ObS gdzie stworzyć plik
   3. tworzy unikalny identyfikator dla pliku
   4. dokonuje transakcji
4. TSVR zwraca identyfikator pliku

Podsumowanie

- Separacja miejsca przechowywania danych od miejsca ich zarządzania. Zarządzanie plikami wykonywane jest w sposób rozproszony przez wiele maszyn. Dane przechowywane są w ObS.
- Cooperative caching – redukcja obciążenia ObS'ów.
- Każdy węzeł może pełnić dowolną funkcję. W przypadku awarii inny węzeł może przejąć rolę węzła, który uległ awarii.
- Uzycie ObS'ów. Umożliwia inteligentne rozkładanie obciążenia na poszczególne ObS'y oraz gwarantuje większe bezpieczeństwo danych – w przypadku nawet uszkodzenia całego systemu plików same dane w obiektach pozostaną nienaruszone.