Pertemuan 13
SISTEM FILE TERDISTRIBUSI

Pokok Bahasan

- Konsep Sistem file terdistribusi
- Arsitektur
- Client server architecture
- Remote access model
- Upload/ download model
- Client base distribute file systems
- Symetrics arsitecture

Konsep Sistem File terdistribusi

Mempertimbangkan bahwa berbagi data merupakan hal mendasar bagi sistem terdistribusi, tidak mengherankan bahwa sistem file terdistribusi menjadi dasar bagi banyak aplikasi terdistribusi. Sistem file terdistribusi memungkinkan banyak proses untuk berbagi data dalam periode waktu yang lama dengan cara yang aman dan andal. Dengan demikian, mereka telah digunakan sebagai lapisan dasar untuk sistem dan aplikasi terdistribusi. Dalam pembahasan ini, menganggap sistem file terdistribusi sebagai paradigma untuk sistem terdistribusi tujuan umum.

ARSITEKTUR

Mempertimbangkan bahwa berbagi data merupakan hal mendasar bagi sistem terdistribusi, tidak mengherankan bahwa sistem file terdistribusi menjadi dasar bagi banyak aplikasi terdistribusi. Sistem file terdistribusi memungkinkan banyak proses untuk berbagi data dalam periode waktu yang lama dengan cara yang aman dan andal. Dengan demikian, mereka telah digunakan sebagai lapisan dasar untuk sistem dan aplikasi terdistribusi. Dalam pembahasan ini, menganggap sistem file terdistribusi sebagai paradigma untuk sistem terdistribusi tujuan umum.

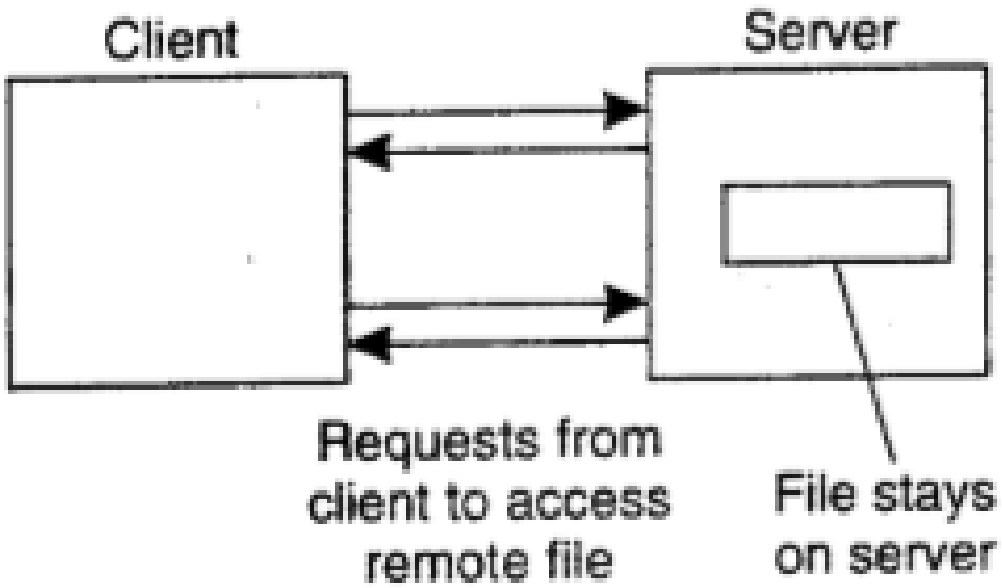
Client-Server Architectures

Banyak sistem file terdistribusi diatur sepanjang arsitektur klien-server. dengan Network File System (NFS) Sun Microsystem menjadi salah satu yang paling banyak digunakan untuk sistem berbasis UNIX. Dapat diambil NFS sebagai contoh kanonik untuk sistem Tile terdistribusi berbasis server di seluruh pembahasan ini. Secara khusus, berkonsentrasi pada NFSv3, versi ketiga NFS yang banyak digunakan.

Ide dasar di balik NFS adalah bahwa setiap server file. memberikan tampilan standar dari sistem file lokalnya. Dengan kata lain, seharusnya tidak masalah bagaimana sistem file lokal diimplementasikan; setiap server NFS mendukung model yang sama. Pendekatan ini telah diadopsi untuk sistem file terdistribusi lainnya juga. NFS dilengkapi dengan protokol komunikasi yang memungkinkan klien untuk mengakses file yang disimpan di server, sehingga memungkinkan kumpulan proses yang heterogen, mungkin berjalan pada sistem operasi dan mesin yang berbeda, untuk berbagi sistem file yang umum.

Remot Access Model

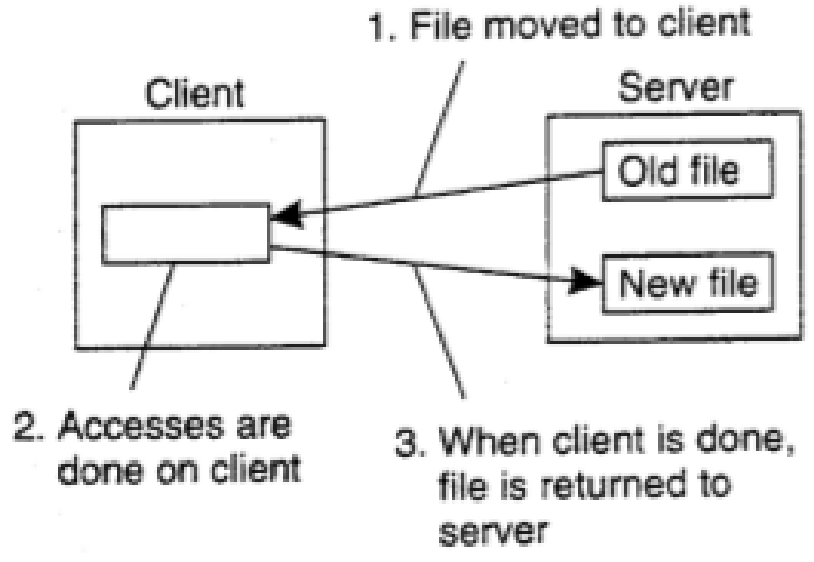
Model yang mendasari NFS dan sistem serupa adalah layanan file jarak jauh. Dalam model ini, klien ditawarkan akses transparan ke sistem file yang dikelola oleh server jauh. Namun, klien biasanya tidak mengetahui lokasi file yang sebenarnya. Sebaliknya, mereka ditawarkan antarmuka ke sistem file yang mirip dengan antarmuka yang ditawarkan oleh sistem file lokal konvensional. Secara khusus, klien hanya ditawarkan antarmuka yang berisi berbagai operasi file, tetapi server bertanggung jawab untuk mengimplementasikan operasi tersebut. Oleh karena itu model ini juga disebut sebagai model akses jarak jauh. Itu ditunjukkan pada Gambar 1.



Gambar 1. The remote access model

Upload/ Download Model

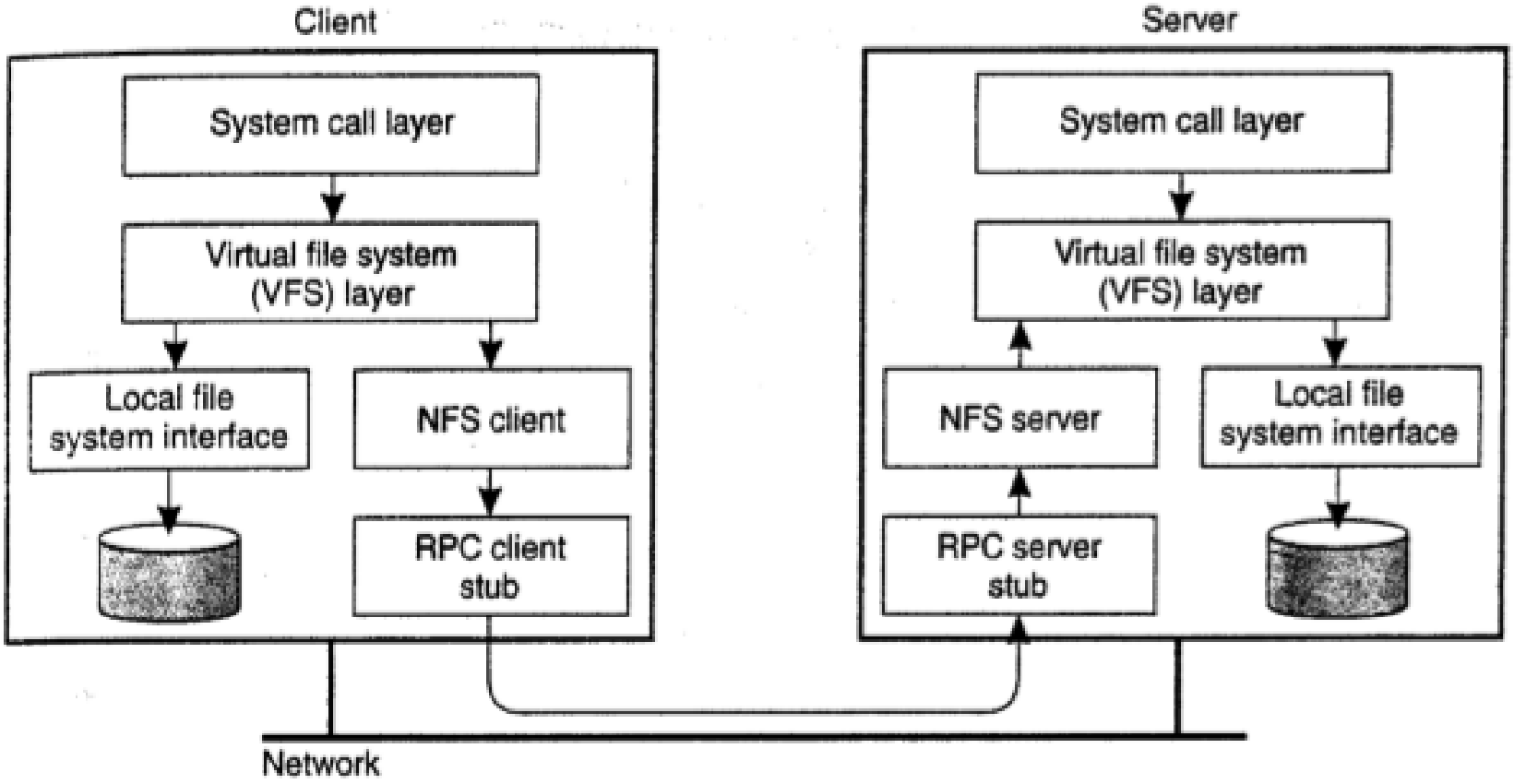
Sebaliknya, dalam model unggah / unduh klien mengakses file secara lokal setelah mengunduhnya dari server, seperti yang ditunjukkan pada Gambar 2. Ketika klien selesai dengan file, itu diunggah kembali ke server lagi sehingga dapat digunakan oleh klien lain. Layanan FTP Internet dapat digunakan dengan cara ini ketika klien mengunduh file lengkap, memodifikasinya, dan kemudian mengembalikannya.



Gambar 2. The upload/download model

NFS telah diimplementasikan untuk sejumlah besar sistem operasi yang berbeda, meskipun versi berbasis UNIX lebih dominan. Untuk hampir semua sistem UNIX modern, NFS umumnya diimplementasikan mengikuti arsitektur berlapis yang ditunjukkan pada Gambar 3. Klien mengakses sistem file menggunakan panggilan sistem yang disediakan oleh sistem operasi lokalnya. Namun, antarmuka sistem file UNIX lokal digantikan oleh antarmuka ke Virtual File System (VFS), yang sekarang merupakan standar de facto untuk berinteraksi dengan berbagai sistem file.

Basic NfS architecture for UNIX systems



Gambar 3. The basic Nf'S architecture for UNIX systems

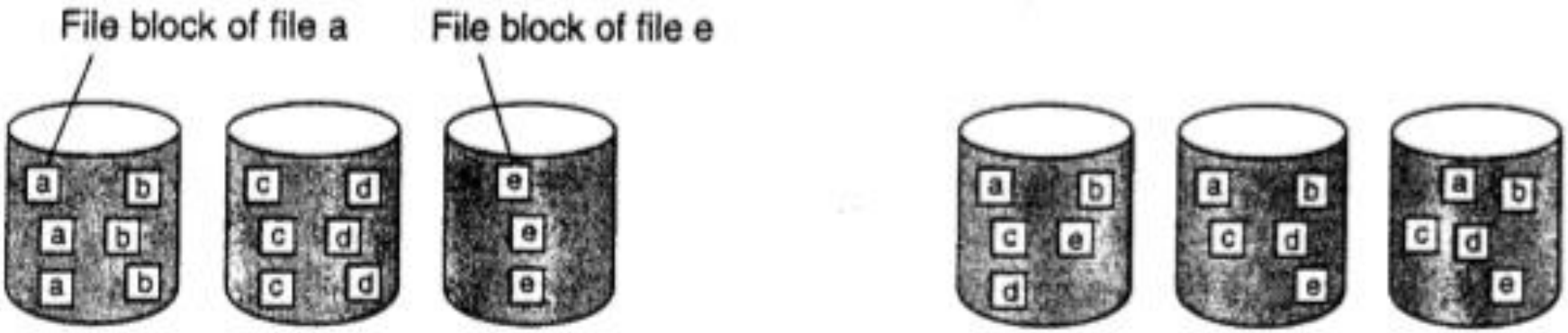
Secara virtual, semua sistem operasi modem menyediakan VFS, dan tidak melakukannya memaksa pengembang untuk menerapkan kembali sebagian besar sistem operasi ketika mengadopsi struktur sistem file yang baru. Dengan NFS, operasi pada antarmuka VFS dapat diteruskan ke sistem file lokal, atau diteruskan ke komponen terpisah yang dikenal sebagai klien NFS, yang menangani penanganan akses ke file yang disimpan di server jauh. Dalam: NFS, semua komunikasi klien-server dilakukan melalui RPC. Klien NFS mengimplementasikan operasi sistem file NFS sebagai RPC ke server. Perhatikan bahwa operasi yang ditawarkan oleh antarmuka VFS dapat berbeda dari yang ditawarkan oleh klien NFS. Seluruh gagasan VFS adalah menyembunyikan perbedaan antara berbagai sistem file.

Di sisi server, ada organisasi serupa. Server NSFS bertanggung jawab untuk menangani permintaan klien yang masuk. RPC rintisan permintaan unmarshals dan server NFS mengubahnya menjadi operasi file VFS biasa yang kemudian diteruskan ke lapisan VFS. Sekali lagi, VFS bertanggung jawab untuk mengimplementasikan sistem file lokal di mana file aktual disimpan.

Keuntungan penting dari skema ini adalah bahwa NFS sebagian besar independen dari sistem file lokal. Pada prinsipnya, itu benar-benar tidak mengetahui apakah sistem operasi pada klien atau server mengimplementasikan sistem file UNIX, sistem file Windows 2000, atau bahkan sistem file MS-DOS. Satu-satunya masalah penting adalah bahwa sistem file ini sesuai dengan model sistem file yang ditawarkan oleh NFS. Misalnya, MS-DOS dengan nama file pendeknya tidak dapat digunakan untuk mengimplementasikan server NFS dengan cara yang sepenuhnya transparan.

Cluster-Based Distributed File Systems

NFS adalah contoh khas untuk banyak sistem file terdistribusi, yang umumnya diatur menurut arsitektur client-server tradisional. Arsitektur ini sering ditingkatkan untuk cluster server dengan beberapa perbedaan. Mempertimbangkan bahwa cluster server sering digunakan untuk aplikasi paralel. Salah satu teknik yang terkenal untuk menyebarkan teknik striping file, dimana satu file didistribusikan di beberapa server. Ide dasarnya sederhana: dengan mendistribusikan file besar di beberapa server, menjadi mungkin untuk mengambil bagian yang berbeda secara paralel. Tentu saja, organisasi semacam itu hanya berfungsi dengan baik jika aplikasi diatur sedemikian rupa sehingga akses data paralel masuk akal. Ini biasanya mengharuskan data yang disimpan dalam file memiliki struktur yang sangat teratur.



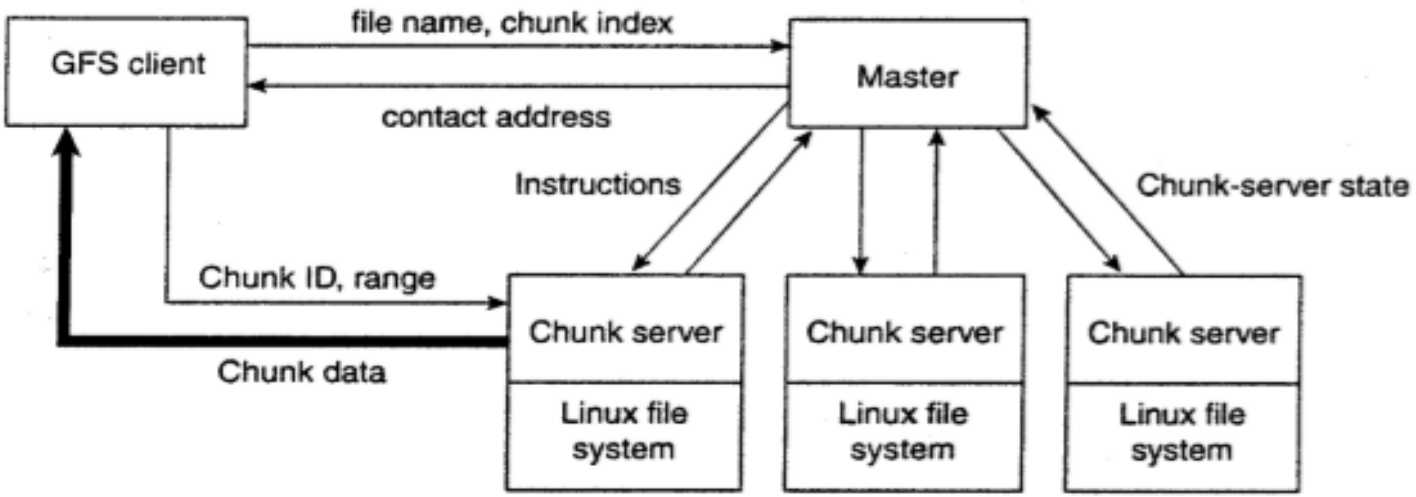
4(a) 4(b)

Gambar 4. (a) distributing whole files across several servers;
(b) striping files for parallel access.

file yang berbeda di server jelas sekali berbeda, tetapi tidak mempartisi file tunggal di beberapa server.

Yang lebih menarik adalah kasus-kasus mengatur sistem file terdistribusi untuk pusat data yang sangat besar seperti yang digunakan oleh perusahaan seperti Amazon dan Google. Perusahaan-perusahaan ini menawarkan layanan kepada klien Web yang menghasilkan pembacaan dan pembaruan ke sejumlah besar file yang didistribusikan secara harfiah puluhan ribu. Dalam lingkungan seperti itu, asumsi tradisional tentang sistem file terdistribusi tidak lagi berlaku. Sebagai contoh, dapat berharap bahwa pada suatu saat akan ada kerusakan komputer. Untuk mengatasi masalah ini, Google, misalnya, telah mengembangkan sendiri sistem file Google (GFS).

File Google cenderung sangat besar, biasanya berkisar hingga beberapa gigabyte, di mana masing-masing berisi banyak objek yang lebih kecil. Selain itu, pembaruan file biasanya dilakukan dengan menambahkan data daripada menimpa bagian file, mengarah pada pembangunan cluster server seperti yang ditunjukkan pada Gambar 5.



Gambar 5. The organization of a Google cluster of servers

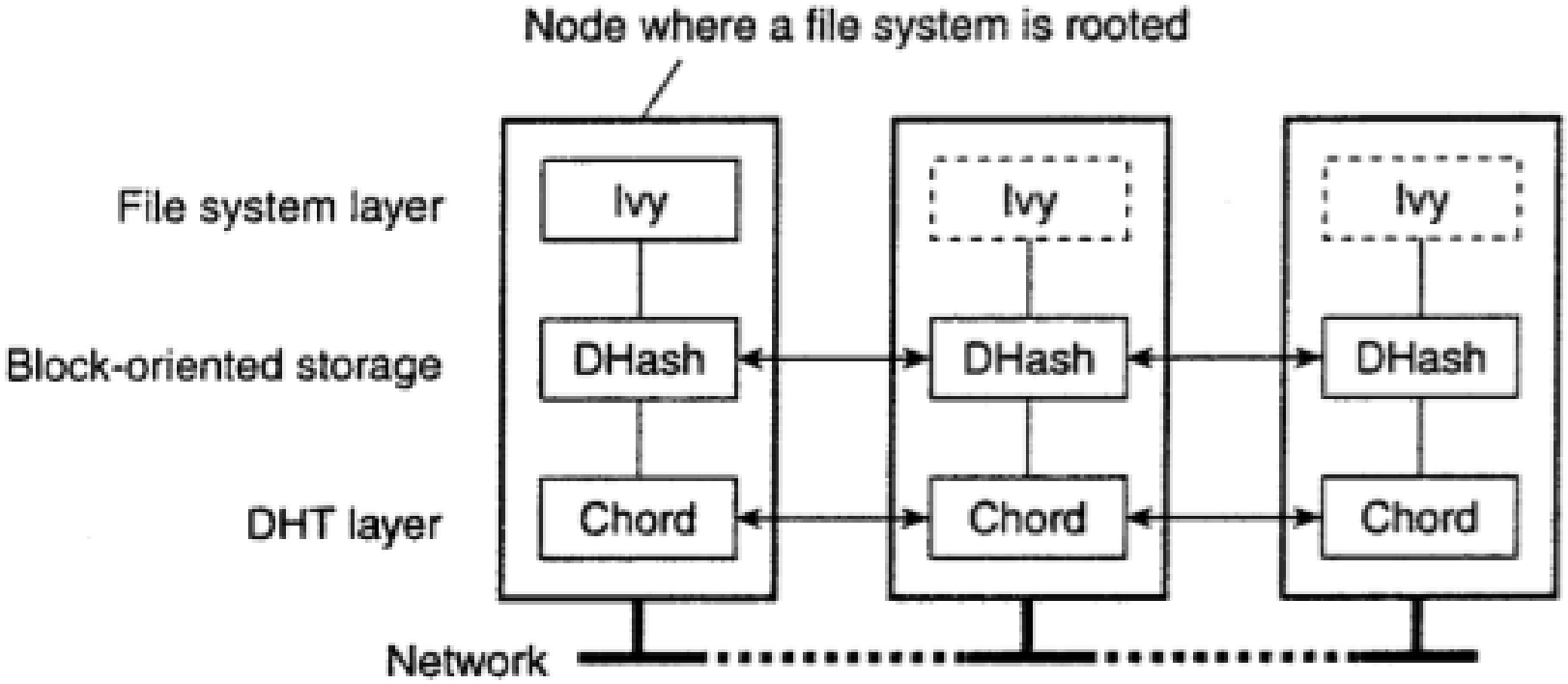
Setiap cluster GFS terdiri dari satu master bersama dengan beberapa server chunk. Setiap file GFS dibagi menjadi potongan masing-masing 64 Mbyte, setelah ini potongan didistribusikan di seluruh apa yang disebut server chunk. Pengamatan penting adalah bahwa master GFS dihubungi hanya untuk informasi metadata. Secara khusus, klien GFS meneruskan nama file dan indeks chunk ke master, mengharapkan alamat kontak untuk chunk. Alamat kontak berisi semua informasi untuk mengakses server chunk yang benar untuk mendapatkan chunk file yang diperlukan.

Untuk tujuan ini, master GFS pada dasarnya mempertahankan ruang nama, bersama dengan pemetaan dari nama file ke potongan. Setiap chunk memiliki pengidentifikasi terkait yang akan memungkinkan server chunk untuk mencarinya. Selain itu, master melacak di mana chunk berada. Bongkahan direplikasi untuk menangani kegagalan, tetapi tidak lebih dari itu. Fitur yang menarik adalah bahwa master GFS tidak berusaha untuk menjaga akun akurat dari lokasi chunk. Sebagai gantinya, kadang-kadang menghubungi server chunk untuk melihat potongan mana yang telah mereka simpan.

Symmetric Architectures

Tentu saja, organisasi yang sepenuhnya simetris yang didasarkan pada teknologi peer-to-peer juga ada. Semua proposal saat ini menggunakan sistem berbasis DHT untuk mendistribusikan data, dikombinasikan dengan mekanisme pencarian berbasis kunci. Perbedaan penting adalah apakah mereka membangun sistem file di atas lapisan penyimpanan terdistribusi, atau apakah seluruh file disimpan pada node yang berpartisipasi.

Sistem mereka pada dasarnya terdiri dari tiga lapisan terpisah seperti yang ditunjukkan pada Gambar 6. Lapisan terendah dibentuk oleh sistem Chord yang menyediakan fasilitas pencarian dasar yang terdesentralisasi. Di tengah adalah lapisan penyimpanan berorientasi blok didistribusikan sepenuhnya. Akhirnya, di atas ada lapisan yang menerapkan sistem file seperti NFS.



Gambar 6. The organization of the Ivy distributed file system

Penyimpanan data di Ivy diwujudkan dengan sistem penyimpanan terdistribusi berbasis blok Chord yang disebut DHash. Intinya, DHash cukup sederhana. Hanya tahu tentang blok data, masing-masing blok biasanya memiliki ukuran 8 KB. Ivy menggunakan dua jenis blok data. Blok hash konten memiliki kunci terkait, yang dihitung sebagai hash aman konten blok tersebut. Dengan cara ini, setiap kali blok dilihat, klien dapat segera memverifikasi apakah blok yang benar telah dicari, atau bahwa versi lain atau rusak dikembalikan.

Selain itu, Ivy juga menggunakan blok kunci publik, yaitu blok yang memiliki kunci publik sebagai kunci pencarian, dan yang isinya telah ditandatangani dengan kunci privat terkait. Untuk meningkatkan ketersediaan, DHash mereplikasi setiap blok B ke k penerus langsung dari server yang bertanggung jawab untuk menyimpan B. Selain itu, mencari blok juga di-cache di sepanjang rute yang diikuti oleh permintaan pencarian.

File diimplementasikan sebagai struktur data terpisah di atas DHash. Untuk mencapai tujuan ini, setiap pengguna memelihara log operasi yang dilakukan pada file, bahwa hanya ada satu pengguna per node sehingga setiap node akan memiliki log sendiri. Log adalah daftar catatan yang tidak dapat diubah, di mana setiap catatan berisi semua informasi yang terkait dengan operasi pada sistem file Ivy. Setiap node menambahkan catatan hanya untuk log sendiri, lokal,. Hanya kepala log yang bisa berubah, dan menunjuk ke catatan yang terakhir ditambahkan. Setiap catatan disimpan dalam blok hash konten yang terpisah, sedangkan kepala alog disimpan dalam blok kunci publik.