SNIFFS: SISTEM BERKAS VIRTUAL UNTUK JARINGAN SOSIAL BERBASIS WEB SERVICE

Kurniawan Lastanto, S.Kom.[1], Dr. Ahmad Ashari, M.Kom.[2]
[1]Guyub Teknologi Nusantara
Jl. K. H. Ahmad Dahlan No. 74 Palembang 30135
[2]Fakultas MIPA, Universitas Gadjah Mada
Sekip Utara Bls, Yogyakarta 55281

E-mail: [1] lastanto@guyub.co.id, [2] ashari@ugm.ac.id

ABSTRAK
Dalam makalah ini diperkenalkan Sniffs (Social Netwok Interface/File System), sebuah sistem berkas virtual untuk jaringan sosial berbasis web service. Sistem berkas ini memungkinkan sembarang program mulai dari utilitas baris perintah sederhana hingga aplikasi desktop yang kompleks untuk mengakses jaringan sosial melalui antarmuka sistem berkas standar. Sniffs menerjemahkan panggilan sistem berkas yang terjadi dan meneruskannya sebagai invokasi terhadap web service yang disediakan oleh penyedia layanan jaringan sosial. Untuk mengakomodasi berbagai web service yang berbeda dari berbagai platform API jaringan sosial, komunikasi spesifik terhadap suatu platform API dilakukan melalui driver-driver konektivitas. Pengelolaan cache diterapkan sebagai upaya untuk mengatasi waktu tunda jaringan dan mengurangi kerja server. Hasil implementasi yang dilakukan dalam penelitian ini menunjukkan bahwa Sniffs mampu menyediakan antarmuka jaringan sosial dalam sistem berkas standar.
Kata Kunci: sistem berkas virtual, jaringan sosial, web service

1. PENDAHULUAN

Keberadaan situs jaringan sosial (*social network site*) seperti Facebook, Friendster, MySpace, dan Google Orkut merupakan fenomena yang sangat menarik. Situs-situs ini mampu menarik jutaan pengguna hanya dalam waktu yang relatif singkat. Akses terhadap situs-situs ini juga telah menjadi kebutuhan sehari-hari bagi sebagian orang. Boyd & Ellison (2007), mendefinisikan situs jaringan sosial (*social network sites*, SNSs) sebagai layanan berbasis web yang memperbolehkan individual untuk membuat sebuah profil publik atau semi-publik dalam sebuah sistem terikat, mengartikulasi daftar pengguna lain yang memiliki hubungan, dan menampilkan dan menelusuri daftar hubunngan dan daftar yang dibuat oleh orang lain dalam sebuah sistem. Sifat dan nomenklatur hubungan ini dapat berbeda dari satu situs dengan situs lainnya.

Munculnya platform API dari para penyedia layanan jaringan sosial yang menyediakan *web service* menjadikan potensi bagi pihak ketiga untuk membangun aplikasi web dan non-web yang dapat digunakan untuk mengakses data jaringan sosial yang disediakan. Namun demikian, setiap penyedia layanan dapat menyediakan sendiri-sendiri API yang tidak kompatibel satu sama lain. Hal ini merupakan suatu permasalahan yang harus dipecahkan untuk dapat membangun aplikasi yang portabel atau mudah diperluas terhadap sembarang API yang disediakan.

Di sisi lain, konsep dasar sistem berkas (*file system*) merupakan salah satu hal yang dipahami oleh hampir semua pengguna komputer. Ketika seseorang mencari berkas (*file*), menelusuri folder, membuka dan menyunting isi berkas maka sebenarnya ia telah memanfaatkan sistem berkas.

Sistem berkas dapat berupa sistem berkas berbasis diska (*disk based file system*) untuk mengelola ruang memori dalam partisi diska lokal, seperti ext3, reiserfs, UFS, VFAT, dan NTFS atau dapat pula berupa sistem berkas jaringan (*network file system*) untuk memudahkan akses terhadap berkas-berkas yang dimiliki komputer lain dalam jaringan, seperti NFS, Coda, AFS (*Andrew file system*), SMB (*Server Message Block*), dan NCP (*Novel NetWare Core Protocol*). Selain itu dapat pula berupa sistem berkas khusus/virtual (*special*/*virtual file system*). Sistem berkas yang disebutkan terakhir ini tidak mengelola ruang diska, baik lokal maupun jarak jauh, contohnya adalah `/proc` (Bovet & Cesati, 2002).

Penelitian yang disampaikan dalam makalah ini mengeksplorasi kemungkinan penyediaan antarmuka jaringan sosial dalam bentuk sistem berkas standar yakni berupa sistem berkas virtual yang dapat diakses oleh sembarang aplikasi lokal melalui pemanfaatan *web service* yang disediakan oleh platform API berbagai penyedia layanan jaringan sosial.

Dengan demikian, ketersediaan antarmuka jaringan sosial dalam bentuk sistem berkas standar memungkinkan pengguna akhir dapat mengakses dan memperbarui data mereka dalam jaringan sosial melalui aplikasi favoritnya, misalnya melihat foto album dengan Gwenview atau Gthumb, melihat daftar teman dengan perintah `ls` atau melalui utilitas manajemen berkas seperti Konqueror, Dolphin, dan Nautilus, serta mencari teman yang berulang tahun

bulan ini dengan perintah find, Kfind, atau Gnome Search for Files (`gnome-search-tool`). Bagi para pengembang, manfaat yang didapat adalah tersedianya alternatif pengembangan aplikasi jaringan sosial hanya dengan memanfaatkan dan memanipulasi sistem berkas standar seperti pembacaan dan penulisan berkas tanpa perlu memahami berbagai *web service* yang disediakan penyedia layanan jaringan sosial. Selain itu, bahasa skrip yang hanya mendukung akses sistem berkas lokal tanpa kemampuan jaringan pun kemudian dapat dimanfaatkan untuk membangun aplikasi berbasis jaringan sosial.

Namun demikian, kondisi Internet tidaklah sama dengan diska lokal. Kecepatan akses data di Internet jauh lebih lambat jika dibandingkan diska lokal. Hal ini menyebabkan perlu adanya suatu strategi agar pengalaman yang dirasakan seorang pengguna ketika menggunakan sistem berkas yang dibangun dalam penelitian ini tidak jauh berbeda dengan pengalaman yang dirasakan ketika pengguna tersebut menggunakan sistem berkas dengan data yang tersimpan dalam penyimpanan lokal.

Dalam penelitian ini, Sniffs diimplementasikan dalam bahasa pemrograman Java dan diujicoba dalam sistem operasi GNU/Linux distribusi Fedora 10 terhadap jaringan sosial Facebook dan Google Orkut.

2. PEMETAAN KONSEP

Dalam situs-situs web jaringan sosial, para pengguna menggunakan browser web untuk mengelola akunnya. Halaman atau bagian halaman tertentu didesain untuk suatu fitur tertentu pula. Hal ini berbeda dengan sistem berkas. Sistem berkas memiliki operasi-operasi yang telah terstandardisasi. Oleh karena itu perlu adanya pemetaan konsep yang mengasosiasikan konsep dalam jaringan sosial dengan sistem berkas.

Pemetaan yang dibuat diusahakan semaksimal mungkin secara intuitif dapat menggambarkan kaitan antara konsep dalam sistem berkas dengan konsep dalam jaringan sosial yang berkorespondensi. Contoh pemetaan konsep ini dapat dilihat pada Tabel 1.

3. DESAIN ARSITEKTUR

Ada tiga hal utama yang menjadi pertimbangan desain arsitektur Sniffs, yakni

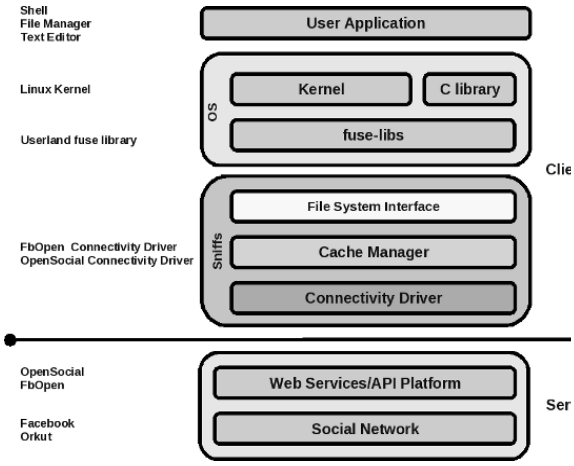
1. Sebagaimana dibahas di Bagian 2, diperlukan pemetaan konsep antara sistem berkas dan jaringan sosial. Selain itu operasi-operasi sistem berkas haruslah dapat dilakukan secara transparan sehingga dapat diakses oleh aplikasi dan utilitas yang telah ada secara langsung tanpa modifikasi.

Tabel 1 Pemetaan Konsep

No.	*Konsep*	*Dalam Situs Web Jaringan Sosial*	*Dalam Sistem Berkas*
1	Penampilan profil pengguna	Pengaksesan halaman/ kotak profil pengguna	Pembacaan berkas-berkas profil pengguna
2	Pengubah-an status pengguna	Pengubahan status melalui form pemutakhiran status	Pengubahan isi berkas status
3	Penampilan daftar album foto pengguna	Pengaksesan halaman/ kotak daftar album foto pengguna	Penampilan isi direktori album foto pengguna
4	Penampilan foto pengguna	Pengaksesan halaman/ kotak foto pengguna	Penampilan berkas-berkas foto pengguna
5	Pembuatan album foto baru pengguna	Pembuatan album foto baru melalui form pembuatan album baru pengguna	Pembuatan direktori album foto baru
6	Penambah-an foto ke dalam album foto pengguna	Pengunggahan (*uploading)* berkas foto ke album yang diinginkan	Penyimpanan/ penyalinan foto ke dalam direktori album foto yang diinginkan
7	Penampilan daftar *note* pengguna	Pengaksesan halaman/kotak daftar *note* pengguna	Penampilan isi direktori lokasi *note* pengguna
8	Penampilan isi *note* pengguna	Pengaksesan halaman/kotak isi *note* pengguna	Penampilan isi berkas *note* pengguna
9	Pembuatan *note* baru pengguna	Pembuatan *note* baru melalui form pengelolaan *note*	Penyimpanan/ penyalinan berkas *note* baru ke dalam direktori lokasi *note* pengguna
10	Penyun-tingan *note* pengguna	Penyuntingan *note* melalui editor *note* pengguna	Penyuntingan berkas *note* dalam direktori lokasi *note* pengguna
11	Penghapus-an *note* pengguna	Penghapusan *note* melalui form pengelolaan *note*	Penghapusan berkas *note* dalam direktori lokasi *note* pengguna
12	Penampilan daftar teman	Pengaksesan halaman/kotak daftar teman	Penampilan isi direktori lokasi teman
13	Penampilan profil teman	Pengaksesan halaman/kotak profil teman	Pembacaan berkas-berkas profil teman
14	Penampilan daftar album foto teman	Pengaksesan halaman/kotak daftar album foto pengguna	Penampilan isi direktori album foto teman
15	Penampilan foto teman	Pengaksesan halaman/kotak foto teman	Penampilan berkas-berkas foto teman

2. Platform API jaringan sosial diakses melalui Internet. Kecepatan lalu lintas data Internet jauh di bawah kecepatan akses diska lokal yang biasa digunakan sebagai media penyimpanan suatu sistem berkas.
3. Platform API yang ditawarkan oleh para penyedia layanan jaringan sosial tidaklah tunggal, maka Sniffs haruslah didesain untuk memungkinkan dukungan terhadap suatu layanan jaringan sosial baru dapat ditambahkan secara bersih melalui API yang disediakan tanpa harus memodifikasi kode sumber Sniffs yang telah ada.

Dengan pertimbangan tersebut, maka Sniffs didesain sebagaimana tampak dalam Gambar 1.



Gambar 1. Arsitektur `Sniffs`

3.1 File System Interface

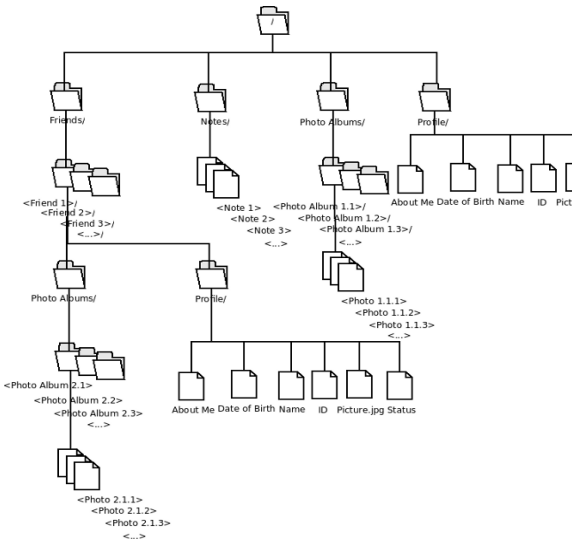
Lapisan *File System Interface* berfungsi menerima panggilan sistem berkas dari sistem operasi. Lapisan ini digunakan untuk menjawab pertimbangan desain arsitektur nomor 1.

Agar dapat dimanfaatkan secara transparan, sistem berkas yang dibuat haruslah dapat menerima akses sistem berkas standar sistem operasi *UNIX-like*. Untuk keperluan ini, FUSE dimanfaatkan untuk menjembatani panggilan sistem (*system call*) sistem berkas standar *UNIX-like* ke Sniffs.

Oleh karena dalam penelitian ini Sniffs diimplementasikan dalam bahasa pemrograman Java, maka untuk dapat berinteraksi dengan kernel melalui FUSE, Sniffs memanfaatkan pustaka fuse-j sebagai binding FUSE dalam bahasa Java.

3.1.1 Desain Hirarki Sistem Berkas

Dalam Sniffs, data jaringan sosial seseorang direfleksikan ke dalam berkas dan direktori. Di bawah titik *mount* (*mount point*), terdapat direktori `Profile`, `Photo Albums`, `Notes`, dan `Friends`. Direktori `Friends` berisi direktori-direktori teman pengguna. Untuk memudahkan navigasi, direktori-direktori ini diberi nama sesuai dengan nama lengkap masing-masing. Namun demikian, apabila terdapat nama yang sama maka ditambahkan ID-nya masing-masing di belakang nama dalam format `Name – ID`. Hal ini diperlukan untuk menjamin bahwa nama direktori masing-masing teman bersifat unik. Hal yang serupa juga diberlakukan pada nama direktori album foto, foto, dan berkas *note*, apabila terdapat nama album foto, foto, dan berkas *note* yang sama, maka ditambahkan ID masing-masing objek yang diwakili. Hirarki ini diilustrasikan oleh Gambar 2.



Gambar 2. Hirarki Sistem Berkas Sniffs

3.1.2 Desain Ijin Akses Sistem Berkas

Sniffs menggunakan ijin akses (*permission*) sistem berkas yang ada dalam sistem *UNIX-like* untuk memberikan informasi tentang hak akses yang dimiliki seorang pengguna terhadap berkas-berkas tersebut.

Oleh karena setiap berkas dan direktori dalam Sniffs berasosiasi dengan objek-objek yang diwakilinya dalam jaringan sosial, maka ijin akses yang dikenakan terhadap setiap berkas dan objek tersebut dikaitkan dengan hak yang diberikan oleh penyedia layanan jaringan sosial.

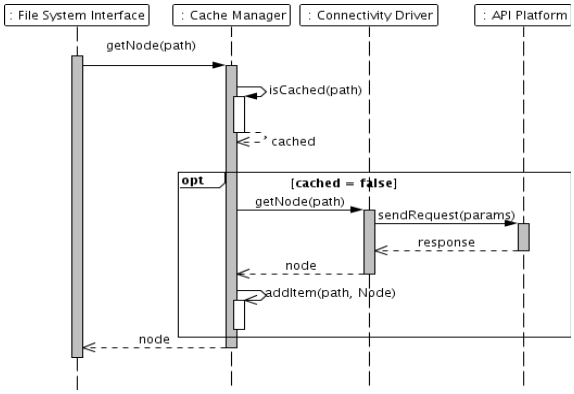
3.2 Cache Manager

Cache Manager dibuat untuk menjawab pertimbangan desain arsitektur nomor 2.

Pengambilan data dari Internet merupakan proses yang sangat mahal jika dibandingkan dengan pengambilan data dari diska lokal. Untuk itu Sniffs didesain untuk menyediakan fungsi penyimpanan sementara dalam memori atau dalam media penyimpanan lokal. Fungsi ini diterapkan dalam lapisan ini. Masa kadaluarsa data yang berada dalam *cache* dapat dikonfigurasi.

Dalam penelitian ini, *Cache Manager* diimplementasikan dengan memanfaatkan ehcache, sebuah pustaka untuk mengelola *cache* dalam bahasa Java.

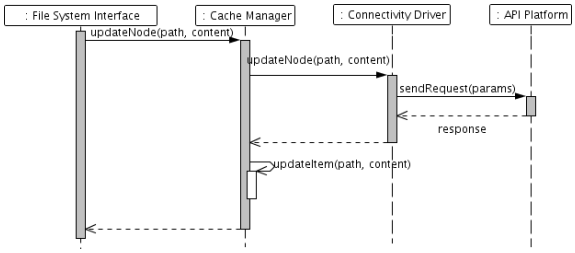
Ketika terjadi permintaan terhadap suatu *node* (berkas atau direktori), maka Sniffs akan mencoba untuk memenuhinya terlebih dahulu dari *Cache Manager*. Apabila node yang dicari dalam *Cache Manager* tidak ditemukan atau kadaluarsa, maka data jaringan sosial yang berkorespondensi dengan *node* tersebut akan diambil atau diambil ulang dari *web service*. Data ini kemudian digunakan untuk memutakhirkan *node* yang dimaksud sekaligus digunakan untuk menjawab permintaan yang terjadi. Gambar 3 mengilustrasikan bagaimana suatu *node* (berkas atau direktori) diambil dari *Cache Manager*.



Gambar 3. *Sequence Diagram* Pengambilan *Node*

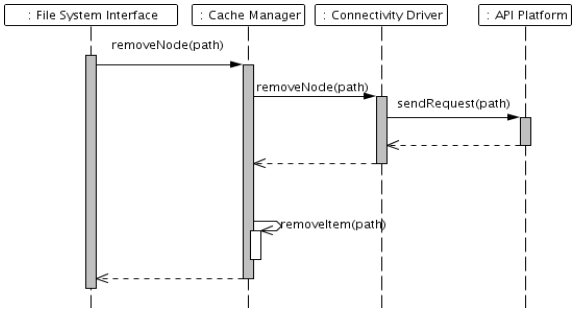
Ketika pengguna melakukan pembaruan data jaringan sosial melalui penambahan atau pembaruan *node*, maka *node* yang telah diperbarui akan digunakan untuk memutakhirkan *Cache Manager* sekaligus memutakhirkan data dalam *web service* melalui *Connectivity Driver* yang digunakan.

Gambar 4 menunjukkan bagaimana ketika suatu *node* (berkas atau direktori) ditambahkan/diperbarui dari *Cache Manager*. Ketika pengguna melakukan penghapusan data jaringan sosial melalui penghapusan node, maka node tersebut akan dihapus dari *cache* dan melalui perantara *Connectivity Driver*, data jaringan sosial yang berkaitan dengan *node* tersebut akan dihapus pula dari basis data penyedia layanan jaringan sosial.



Gambar 4. *Sequence Diagram* Pemutakhiran *Node*

Gambar 5 menunjukkan bagaimana ketika suatu *node* (berkas atau direktori) dihapus dari *cache*.



Gambar 5. *Sequence Diagram* Penghapusan *Node*

3.3 Connectivity Driver

Connectivity Driver berfungsi memetakan API *web service* penyedia layanan jaringan sosial ke API Sniffs. Driver ini akan berbeda untuk setiap *API Platform* yang berbeda. Lapisan ini dibuat untuk menjawab pertimbangan desain arsitektur nomor 3.

Apabila diinginkan untuk memberi dukungan terhadap suatu *API Platform* baru maka pengembang hanya perlu membuat *connectivity driver* yang mendukung *API Platform* baru tersebut dan menempelkannya ke Sniffs secara bersih melalui API yang disediakan tanpa harus memodifikasi kode sumber Sniffs yang telah ada.

Sebagai contoh dalam penelitian ini, selain dibuat driver untuk mendukung Facebook Open Platform (FBOpen) juga dibuat driver untuk mendukung OpenSocial.

3.3.1 Implementasi Driver Facebook Open

Driver Facebook Open diimplementasikan secara langsung memanfaatkan *web service* yang disediakan oleh platform API Facebook Open.

Ketika terdapat permintaan data dari *File System Interface* melalui *Cache Manager*, permintaan ini akan diteruskan oleh driver ini ke Facebook API Platform (Facebook Developers, 2008). Pada setiap pemanggilan *method* yang disediakan oleh platform, Sniffs menyertakan parameter `format` dengan nilai JSON, untuk memperoleh respons dalam format JSON.

3.3.2 Implementasi Driver OpenSocial

Driver OpenSocial diimplementasikan secara langsung memanfaatkan *web service* yang disediakan oleh OpenSocial Platform API (Google, 2008). Pada setiap permintaan terhadap *endpoint* REST yang disediakan, Sniffs meminta jawaban dalam format JSON. JSON dipilih karena enkapsulasi data dalam format ini lebih lebih ringkas dari pada dalam format XML maupun atomPub. Dengan demikian *bandwidth* yang diperlukan pun dapat lebih rendah jika dibandingkan dengan menggunakan format XML maupun atomPub.

Oleh karena dukungan implementasi OpenSocial dari para penyedia layanan jaringan sosial saat ini masih sangat terbatas, maka implementasi Driver OpenSocial lebih dimaksudkan untuk menunjukkan

bahwa Sniffs dapat diperluas untuk mendukung platform baru melalui penambahan driver konektivitas.

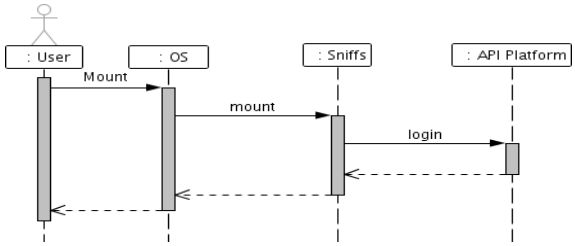
4. OPERASI SISTEM BERKAS

4.1 Proses *Mounting*

Sebagaimana dalam sistem berkas konvensional, pengguna melakukan *mounting* untuk menggabungkan ruang nama Sniffs ke sistem berkas utama. Dalam Sniffs proses *mounting* sejatinya merupakan proses membaca berkas konfigurasi, login ke *API Platform*. Setelah proses *mounting* dilakukan, Sniffs telah tersedia di bawah titik *mount* (*mount point*) dan siap menerima perintah sebagai mana sistem berkas lokal.

Isi berkas konfigurasi berisi informasi tentang parameter yang diperlukan untuk mengakses suatu jaringan sosial tertentu. Oleh karena itu, pasangan kunci dan nilai yang dibutuhkan bervariasi dari satu jaringan sosial ke jaringan sosial yang lain.

Proses *mounting* ditampilkan oleh Gambar 6.

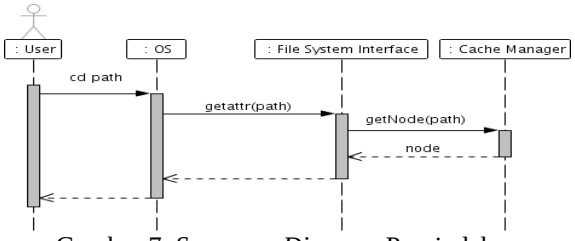


Gambar 6. *Sequence Diagram* Proses *Mounting*

4.2 Perpindahan Direktori

Aktivitas perpindahan direktori menyebabkan pemanggilan `getattr()` dengan parameter berupa *path* direktori tujuan. Pemanggilan ini dilakukan oleh sistem operasi untuk mendapatkan informasi mengenai atribut direktori yang menjadi parameternya. Untuk menjawab panggilan ini, lapisan *File System Interface* meneruskan parameter ini melalui `getNode()` di *Cache Manager* untuk mendapatkan jawaban yang diperlukan.

Proses ini ditampilkan pada Gambar 7. Oleh *Cache Manager*, pemanggilan `getNode()` dapat dipenuhi melalui data yang telah ada di *Cache Manager* sebelumnya atau jika belum ada akan dipenuhi melalui permintaan ke *API Platform* melalui driver yang digunakan (lihat Gambar 3).



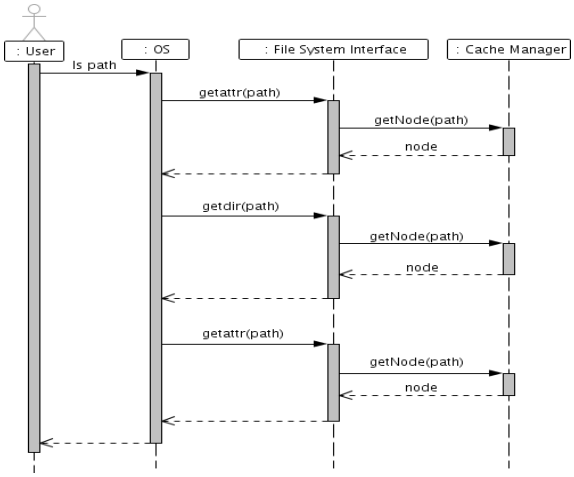
Gambar 7 *Sequence Diagram* Perpindahan Direktori

4.3 Penampilan Isi Direktori

Untuk mengetahui daftar teman atau berkas profil apa saja yang ada dapat dilakukan dengan melihat isi direktori. Operasi ini dapat dilakukan melalui perintah `ls` atau membuka direktori yang diinginkan dengan pengelola berkas seperti `Dolphin` dan `Konqueror` pada KDE dan `Nautilus` pada Gnome.

Sistem Operasi mengetahui isi direktori melalui `getdir()`. Ketika even ini terjadi, *File System Interface* akan menjawabnya dengan mengambil informasi isi direktori melalui pemanggilan `getNode()` pada *Cache Manager* dengan parameter yang diambil dari parameter yang dilewatkan melalui `getdir()`.

Untuk panggilan `getattr()` dari Sistem Operasi dan metode `getNode()` yang tersedia di *Cache Manager* dapat dilihat pada Bagian 3.2.



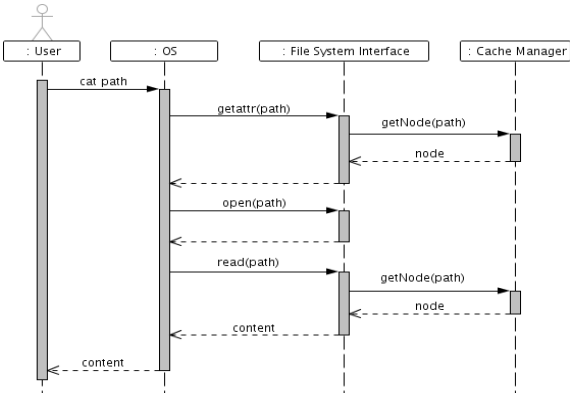
Gambar 8. *Sequence Diagram* Penampilan Isi Direktori

Sequence diagram penampilan isi direktori dapat dilihat pada Gambar 8.

4.4 Pembacaan Berkas`

Dalam Sniffs, aktivitas dalam jaringan sosial seperti melihat profil dan foto dilakukan melalui pembukaan dan pembacaan berkas. Perintah yang dilakukan pengguna dapat berupa baris perintah seperti `cat Status` atau melihat foto dengan aplikasi `Gwenview` pada `KDE` atau `Gthumb` pada `Gnome`.

Proses pembacaan berkas diilustrasikan oleh Gambar 9. Proses ini diawali dengan pemanggilan `getattr()` dan `open()` oleh Sistem Operasi untuk mendapatkan atribut berkas dan membukanya. Sebagai parameter, *path* berkas disertakan dalam pemanggilan ini. Setelah itu, `read()` dipanggil dengan parameter yang sama dilakukan, *File System Interface* kemudian akan meneruskan parameter ini ke *Cache Manager* melalui `getNode()` untuk mendapatkan berkas yang diinginkan. Isi berkas ini kemudian dikembalikan ke OS melalui buffer yang telah disiapkan.

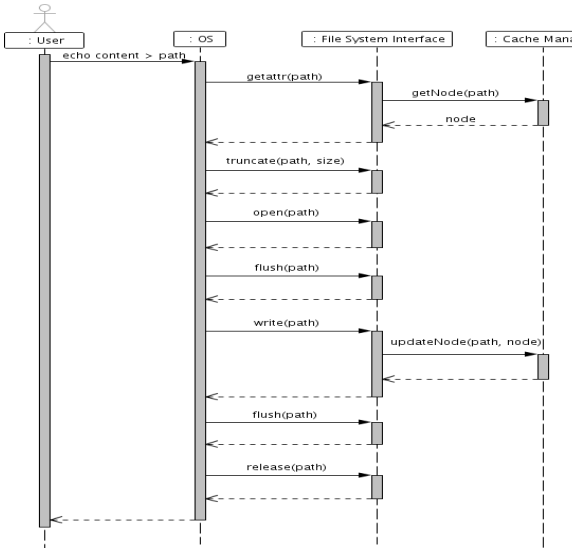


Gambar 9 *Sequence Diagram* Pembacaan Berkas

4.5 Penulisan Berkas

Pembaruan Status dalam layanan jaringan sosial dengan penyunting teks atau melalui perintah *shell* `echo "Status Baru" > Status` merupakan contoh aktivitas yang melibatkan proses penulisan ke berkas.

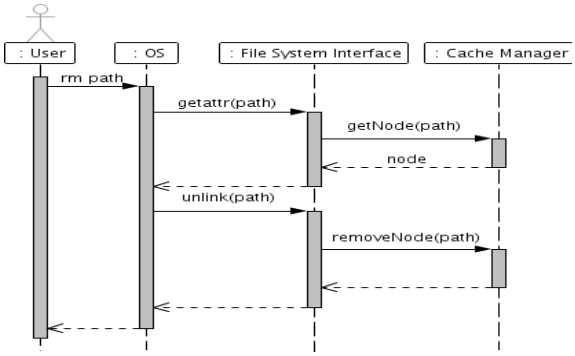
Proses penulisan ke berkas ditampilkan dalam Gambar 10. Proses ini diawali dengan pemanggilan `getattr()`, `truncate()`, `open()` dan `flush()` oleh Sistem Operasi untuk mendapatkan atribut berkas, memotong berkas ke ukuran baru, membuka, dan menggelontor *buffer* yang digunakan. *Buffer* yang berisi nilai yang akan ditulis kemudian dituliskan ke berkas melalui `write()`, *File System Interface* akan meneruskan proses ini ke dalam *cache* melalui `updateNode()` untuk memperbarui berkas di *API Platform* dan di internal *Cache Manager*.



Gambar 10 *Sequence Diagram* Penulisan Berkas

4.6 Penghapusan Berkas

Dalam jaringan sosial, sejumlah informasi dapat dihapus. Sniffs memanfaatkan even panggilan `unlink` untuk mendukung konsep ini. Gambar 11 menggambarkan operasi ini.



Gambar 11 *Sequence Diagram* Penghapusan Berkas

5. PENGUJIAN

Pengujian dilakukan dengan melakukan penampilan isi direktori, pembuatan direktori, pembacaan berkas, penulisan berkas, maupun penghapusan berkas untuk berkas/direktori yang diuji. Bisa tidaknya suatu berkas untuk dibaca, ditulis, atau dihapus dan suatu direktori dibuat atau dihapus tergantung dari iijin akses yang diterapkan terhadap berkas atau direktori tersebut berdasarkan spesifikasi *web service* pada platform API jaringan sosial yang bersangkutan.

Pengujian ini dilakukan dengan berbagai program baris perintah dan aplikasi grafikal lalu dibandingkan dengan kondisi sesungguhnya atau efek yang terjadi pada situs web jaringan sosial yang bersangkutan.

6. KESIMPULAN

Sniffs dapat menyediakan antarmuka jaringan sosial dalam bentuk sistem berkas standar yang dapat diakses oleh sembarang aplikasi lokal melalui pemanfaatan *web service* yang disediakan oleh platform API berbagai penyedia layanan jaringan sosial. Akomodasi terhadap berbagai platform API yang berbeda dimungkinkan melalui pemisahan komponen yang berfungsi dalam komunikasi spesifik terhadap suatu platform API dalam bentuk driver-driver konektivitas. Dukungan terhadap platform API baru dapat dilakukan dengan penambahan driver.

PUSTAKA

Bovet, P. & Cessati, D (2002). *Understanding Linux Kernel*. Beijing: O'Reilly.

Boyd, D. M. & Ellison, N. (2007). *Social Network Sites: Definition, History, and Scholarship, Journal of Computer-Mediated Communication*, 13(1). Diakses pada 31 Desember 2008 dari http://jcmc.indiana.edu/vol13/issue1/boyd.ellison.html.

Facebook Developers (2008). *API*. Diakses pada 31 Desember 2008 dari http://wiki.developers.facebook.com/index.php/API.

Google (2008). *OpenSocial API Documentation*, Diakses pada 31 Desember 2008 dari http://code.google.com/intl/id/apis/opensocial/docs/.