



KEMENTERIAN PENDIDIKAN DAN KEBUDAYAAN
UNIVERSITAS BRAWIJAYA
FAKULTAS TEKNIK
JURUSAN TEKNIK ELEKTRO
Jalan MT Haryono 167 Telp & Fax. 0341 554166 Malang 65145

KODE
PJ-01

**PENGESAHAN
PUBLIKASI HASIL PENELITIAN SKRIPSI
JURUSAN TEKNIK ELEKTRO
FAKULTAS TEKNIK UNIVERSITAS BRAWIJAYA**

NAMA : GIAN GIOVANI
NIM : 0910630062- 63
PROGRAM STUDI : TEKNIK REKAYASA KOMPUTER
**JUDUL SKRIPSI : ON-DEMAND AUDIO STREAMING MENGGUNAKAN
METODE PEER TO PEER BLOCK SCHEDULING**

TELAH DI-REVIEW DAN DISETUJUI ISINYA OLEH:

Pembimbing 1

Pembimbing 2

Adharul Muttaqin, ST., MT.
NIP. 19760121 200501 1 001

Muhammad Aswin, Ir., MT.
NIP. 19640626 199002 1 001

**ON-DEMAND AUDIO STREAMING MENGGUNAKAN METODE
PEER TO PEER BLOCK SCHEDULING**

Publikasi Jurnal Skripsi



Disusun Oleh :

GIAN GIOVANI

NIM : 0910630062 - 63

**KEMENTERIAN PENDIDIKAN DAN KEBUDAYAAN
UNIVERSITAS BRAWIJAYA
FAKULTAS TEKNIK
MALANG
2013**

ON-DEMAND AUDIO STREAMING MENGGUNAKAN METODE PEER TO PEER BLOCK SCHEDULING

Gian Giovani.¹, Adharul Muttaqin, ST., MT.², Muhammad Aswin, Ir., MT.²

¹Mahasiswa Teknik Elektro Univ. Brawijaya, ²Dosen Teknik Elektro Univ. Brawijaya

Jurusan Teknik Elektro Fakultas Teknik Universitas Brawijaya

Jalan MT. Haryono 167, Malang 65145, Indonesia

E-mail: wejick@gmail.com

Abstract— Audio streaming is process to play music files from network before the download process is finished, now many people use audio streaming. While so much people use audio streaming, new problem arise in conventional centralized audio streaming. Centralized audio streaming need more resource to serve more client. In Peer to Peer Block Scheduling Audio Streaming Block Scheduling Method there are no centralized server to serve client, however load is distributed equally to another client. Peer to Peer Block Scheduling Audio Streaming is implemented with TCP protocol by using Socket API. Testing routine show that client request is distributed almost equally to every available client in system.

Index Terms— Audio Streaming, Peer to Peer, TCP, Socket.

Abstrak— Audio streaming adalah memainkan berkas suara lewat jaringan sebelum berkas sepenuhnya selesai diunduh, saat ini semakin banyak yang memanfaatkan audio streaming. Makin banyaknya orang yang menggunakan audio streaming menimbulkan masalah pada penerapan sistem audio streaming konvensional yang terpusat. Dengan sistem terpusat maka sumberdaya yang dibutuhkan pelayan (server) pusat harus makin besar mengikuti perkembangan pengguna. Pada audio streaming peer to peer metode block scheduling tidak ada pelayan (server) terpusat yang terbebani, melainkan beban didistribusikan pada pengguna itu sendiri. Audio streaming Peer to Peer metode block scheduling di Implementasikan menggunakan protokol TCP dengan memanfaatkan API Socket. Dari hasil pengujian beban pelayanan audio streaming terbagi hampir sama rata pada sistem audio streaming peer to peer metode block scheduling.

Kata Kunci— Audio Streaming, Peer to Peer, TCP, Socket.

I. PENDAHULUAN

Internet telah berkembang menjadi media penyebaran informasi yang banyak digunakan oleh pengguna 32% dari populasi dunia [1]. Salah satu layanan yang banyak digunakan adalah *audio streaming*, yaitu memutar langsung berkas audio sebelum proses pengunduhan lewat jaringan selesai dilakukan[2].

Implementasi sistem audio streaming yang banyak digunakan adalah sistem terpusat, di mana terdapat satu server melayani banyak pengguna. Pada sistem ini terjadi dapat terjadi masalah bila penggunanya semakin banyak, semakin pengguna membutuhkan sumberdaya

yang semakin besar. Sumberdaya tersebut antara lain *bandwidth*, kekuatan komputasi dan media penyimpanan.

Saat ini berkembang model peer to peer yaitu Model berbagi sumber daya yang menyediakan sebuah solusi arsitektur yang atraktif untuk aplikasi dengan bandwidth terbatas. peer-peer menggunakan bandwidth unggah mereka untuk meredistribusikan konten yang telah didownload sebelumnya, mengurangi kebutuhan dan beban pada pelayan (server)[3]. Dengan Peer to Peer beban pada *server* pada sistem tersentralisasi dapat terbagi pada pengguna.

Penelitian dalam skripsi ini akan membahas penerapan *block scheduling* pada sistem audio streaming Peer to Peer. Pengujian dilakukan untuk mendapatkan pengaruh sistem pada *delay* pemutaran dan seberapa besar penyebaran beban pada pengguna.

II. TINJAUAN PUSTAKA

A. Streaming

Pengaksesan berkas audio maupun video melalui jaringan dapat dilakukan seperti memperlakukan berkas teks yaitu dengan mengunduh dan membuka; tetapi kebanyakan pengguna lebih suka langsung memutar berkas bersamaan dengan proses mengunduh berlangsung, hal ini biasa disebut [2]. Terdapat dua pendekatan dalam implementasi streaming, yaitu berbasis Unicast dan berbasis Multicast selain itu juga ada metode implementasi peer to peer.

Streaming Unicast

Dalam pendekatan ini setiap pengguna yang mengakses akan membuat satu hubungan (stream) unicast baru. Saluran unicast adalah sambungan dari titik ke titik dimana terdapat satu penerima dan satu pengirim dalam saluran itu [5]. Implementasi tersentralisasi adalah implementasi berbasis unicast yang banyak digunakan.

Streaming Multicast

Multicast streaming lebih mudah diterapkan pada live streaming di mana setiap pengguna tersinkronisasi: setiap pengguna mendapatkan bagian yang sama dari sebuah saluran dalam satu waktu. Untuk On-Demand streaming di mana setiap pengguna tidak tersinkronisasi, beberapa teknik telah diusulkan. Salah satu ide utama untuk menggunakan pendekatan multicast adalah patching. Dalam teknik ini pengguna diperbolehkan bergabung dalam sesi multicast yang telah berjalan. Sebagai tambahan, pengguna juga

membuat hubungan unicast dengan server untuk mendapatkan bagian dari berkas yang telah terlewat dalam sesi multicast. Penggunaan teknik patching mengharuskan pengguna membuat beberapa saluran dalam satu periode patching, berarti pengguna harus memiliki paling tidak dua kali lebih besar dari pada laju streaming (streaming rate) [6].

B. Streaming Berbasis peer-to-peer

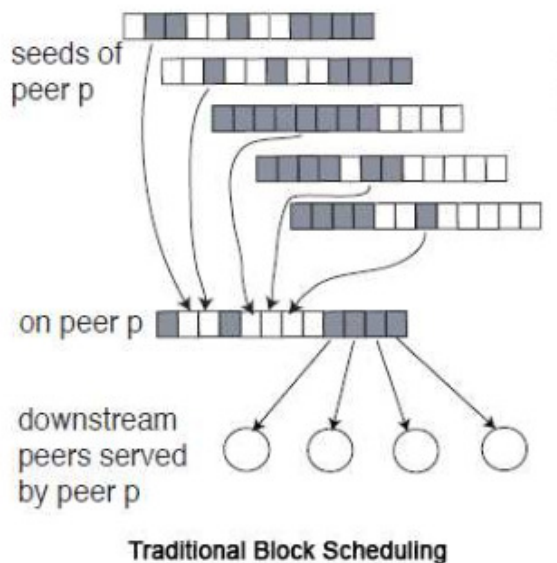
Selain implementasi berbasis unicast dan multicast yang lebih umum, terdapat juga satu implementasi yang saat ini mulai mendapat perhatian yaitu peer-to-peer. Ide kunci dari arsitektur ini adalah pengguna (disebut peer) membagikan sebagian sumberdayanya kepada sistem, sehingga kapasitas sistem secara keseluruhan meningkat dan dapat melayani pengguna lebih banyak [6].

Dalam sistem peer-to-peer setiap permintaan yang dilakukan oleh peer dilayani oleh supplying peer tertentu. Selanjutnya setelah menerima data, peer tersebut menyimpannya pada media penyimpanan lokal sehingga dapat menjadi supplying peer baru untuk peer lain [4].

C. Algoritma Block Scheduling

Peer-to-peer pada dasarnya adalah konsep hubungan langsung antara entitas dalam jaringan komputer di mana posisi keduanya sama yaitu saling melayani dan dilayani. Namun dalam implementasi streaming berbasis peer-to-peer digunakan prinsip block scheduling. Pada streaming berbasis peer-to-peer digunakan *block scheduling* untuk menangani pengiriman block dalam sistem secara efisien dan optimal.

Dalam block scheduling berkas media dipartisi menjadi beberapa blok dengan ukuran yang sama, setiap block memiliki nomor urutan yang unik. Selanjutnya setiap peer memutar blok-blok yang ada sesuai dengan urutan dan dalam kecepatan sesuai dengan laju streaming [3].



Gambar 1. Ilustrasi Algoritma Block Scheduling (Sumber :Le Chang)

III. PERANCANGAN DAN IMPLEMENTASI PERANGKAT LUNAK

Sistem Audio Streaming Peer to Peer Metode *Block Scheduling* diimplementasikan menjadi 2 perangkat lunak yaitu: Hub dan NodePeer. Hub menghimpun dan menyediakan informasi tentang berkas yang tersedia dalam jaringan dan peer lain yang aktif. NodePeer melakukan proses streaming audio dengan meminta informasi berkas yang tersedia ke Hub, melakukan proses permintaan berkas ke peer lain dan memutar berkas audio. NodePeer juga menerima dan melayani permintaan berkas.

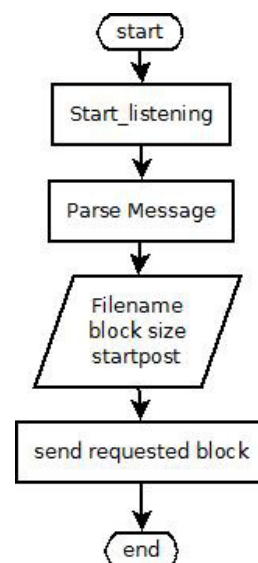
A. Hub

Hub memiliki beberapa fungsi yaitu :

1. Menerima Informasi berkas yang tersedia dari NodePeer.
2. Menerima Informasi status keaktifan NodePeer.
3. Memberikan Informasi berkas yang tersedia pada tiap NodePeer kepada NodePeer.

Pada dasarnya Hub menerima perintah dari NodePeer dan membalasnya sesuai dengan perintah yang diterima. Perintah dan balasan itu disampaikan lewat socket TCP berupa pesan teks polos (plain text). Gambar 2 menunjukkan alur kerja Hub saat mendapatkan perintah dari NodePeer. Berikut ini adalah daftar perintah yang dapat dikirim oleh peer beserta keterangan fungsinya :

1. FL : Untuk mendapatkan daftar berkas
2. FD : Untuk mendapatkan detail berkas beserta peer yang tersedia.
3. NA : Memberitahu status keaktifan peer.
4. UP : Memperbarui catatan blok yang tersedia dalam peer.



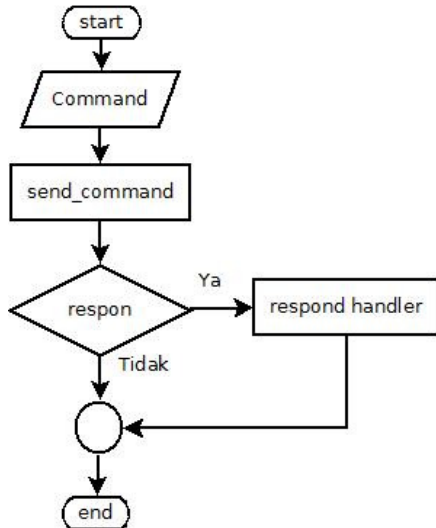
Gambar 2. Alur kerja Hub saat menerima perintah

B. NodePeer

NodePeer terdiri dari empat komponen yang memiliki fungsi masing-masing, yaitu :

1. RedToHub

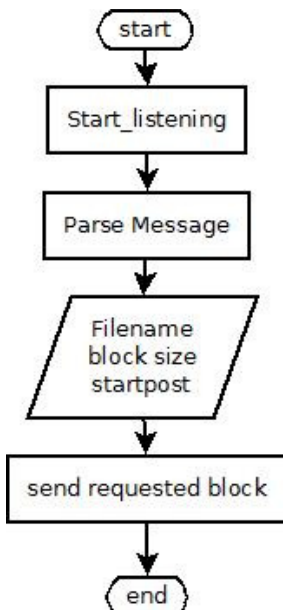
Komponen yang menghubungkan antara NodePeer dengan Hub. RedToHub menjaga informasi antara lain daftar berkas yang tersedia di jaringan peer to peer dan daftar host yang siap melayani permintaan suatu file. Gambar 2 menunjukkan alur kerja saat RedToHub mengirim perintah ke Hub dan menerima respon dari Hub.



Gambar 2. Alur kerja RedToHub saat mengirim perintah dan menerima respon dari Hub

2. NodePeerServ

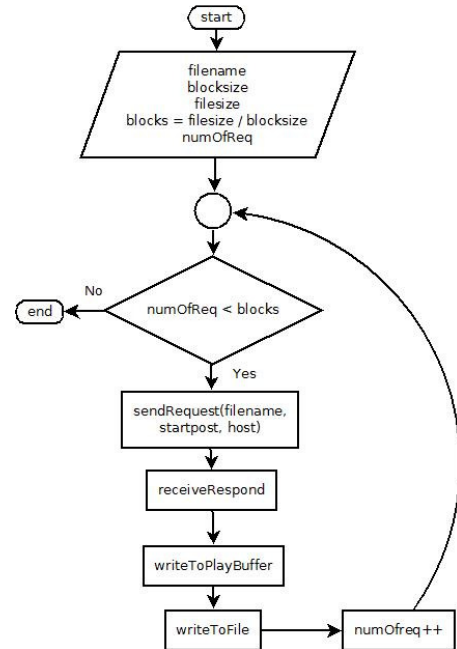
Berfungsi untuk melayani permintaan blok berkas suara dari NodePeer lain. Ketika suatu NodePeer mengirimkan permintaan blok berkas maka AsynchRedServ akan melayani permintaan itu. Gambar 3 menunjukkan saat NodePeerServ menerima dan melayani permintaan.



Gambar 3. Alur kerja Hub saat menerima perintah

3. NodePeerStream

Berfungsi sebagai pengatur proses pengunduhan berkas dari berbagai host nodePeer yang siap untuk mendapat permintaan. AsynchRedStream memastikan berkas yang diunduh untuk berjalan dengan lancar sehingga tidak terjadi penundaan dalam proses streaming. Gambar 4 menunjukkan alur kerja saat streaming dilakukan.



Gambar 4. Alur kerja Hub saat menerima perintah

4. BassPlayer

BassPlayer memiliki fungsi untuk memainkan stream yang sedang berjalan sesuai dengan kondisi stream. Bila suatu blok berkas suara yang seharusnya dimainkan belum tersedia maka BassPlayer akan memberi waktu jeda hingga AsynchRedStream menyelesaikan proses pengunduhan.

IV. PENGUJIAN DAN PEMBAHASAN

A. Total Delay per Playback

Total Delay per Playback adalah jumlah waktu penundaan yang terjadi pada saat pemutaran / streaming berkas suara. Penundaan terjadi karena blok berkas yang seharusnya dimainkan belum sampai, sehingga dibutuhkan waktu untuk menyelesaikan proses pengunduhan berkas tersebut.

Pengujian dilakukan dengan melakukan streaming berkas audio pada 10 komputer, selama streaming dilakukan pencatatan delay. Pencatatan delay dilakukan menggunakan timer pada thread yang berbeda dari program utama dan berjalan dengan interval 100 ms. Timer akan mencatat bila terjadi penundaan saat proses streaming. Pada peer ke-8 tidak dilakukan test karena merupakan host yang mengunggah berkas pertama kali.

Tabel 1. Hasil pengujian delay pemutaran berkas

No	Berkas	Bitrate (Kbps)	Sample rate (Hz)	Panjang	Rata-rata Delay
1	Satu	192	44.100	03:29	0
2	Dua	256	44.100	03:13	0

Tabel 2. Hasil pengujian throughput

No	Rata-rata paket per detik	Rata-rata besar paket(byte)	Throughput (Mbit/s)
1	845,156	700,490	4,736

Dari hasil pengujian tidak ditemukan *delay* pada saat *streaming* berlangsung. Hal ini diakibatkan *throughput* jaringan lebih besar daripada *Bitrate* berkas suara.

B. Total Ignored Transmit Session per Playback dan penyebaran pembagian beban saat streaming

Pengujian Total Ignored Transmit Session per Playback dilakukan dengan cara mencatat tiap sesi yang gagal setelah melewati waktu yang ditentukan (time out), time out yang digunakan adalah 100 ms. Pada peer ke-8 tidak dilakukan test karena merupakan host yang mengunggah berkas pertama kali.

Pengujian penyebaran pembagian beban dilakukan dengan mencatat dari mana sebuah peer mendapatkan blok dan jumlah blok yang diminta dari tiap peer. Pencatatan dilakukan dengan membuat sebuah struktur data dalam program yang menampung data-data yang dibutuhkan.

Tabel 3. Rata-rata Jumlah sesi yang di acuhkan

No	Berkas	Rata-rata request yang di acuhkan
1	Satu	0
2	Dua	0

Tabel 4. Persebaran permintaan oleh host ke-10

No	Blok pada Peer ke-									Total
	1	2	3	4	5	6	7	8	9	
1	66	66	67	67	66	66	-	67	66	531
2	-	-	71	70	70	70	-	71	70	422

Dari hasil pengujian pada Tabel 3 terlihat bahwa tidak ada permintaan dari peer yang diacuhkan, hal ini berarti tidak ada peer yang terlalu sibuk untuk melayani permintaan peer lain. Ketidak adaan peer yang terlalu sibuk melayani permintaan dikarenakan penyebaran beban yang merata pada peer yang ada dalam sistem seperti terlihat pada Tabel 4.

C. Perbandingan Berkas Suara Unduhan

Untuk mengetahui adanya perbedaan antara berkas sumber dengan berkas yang telah diunduh maka dibandingkan kedua berkas tersebut. Pengujian

menggunakan aplikasi WinMerge untuk melihat perbedaan pada berkas binary.

Tabel 4. Perbandingan bit berkas hasil streaming dengan berkas asli

No	Peer	Perbedaan			
		Satu		Dua	
		Byte	Persen (%)	Byte	Persen (%)
1	1	422	0.0084	531	0.0855
2	2	422	0.0084	529	0.0852
3	3	420	0.0083	521	0.0853
4	4	420	0.0083	531	0.0855
5	5	422	0.0084	531	0.0855
6	6	422	0.0084	530	0.0085
7	7	420	0.0083	521	0.0853
8	9	422	0.0084	530	0.0085
9	10	424	0.0084	534	0.0086

Angka pada kolom perbedaan berarti banyaknya titik perbedaan dalam satu berkas binary. Merujuk pada tabel 5.6 dapat terlihat bahwa perbedaan hampir sama dengan jumlah blok yang diminta. Perbedaan disebabkan oleh proses pengiriman berkas lewat jaringan tidak dilakukan rutin koreksi error.

V. PENUTUP

A. Kesimpulan

Dari proses perancangan, implementasi dan pengujian audio streaming peer to peer metode block scheduling didapat kesimpulan antara lain.

1. Audio streaming peer to peer metode block scheduling merupakan salah satu cara melakukan streaming berkas audio melalui jaringan komputer.
2. Sistem audio streaming peer to peer metode block scheduling memiliki 2 bagian aplikasi yaitu Hub sebagai tracker dan NodePeer sebagai peer.
3. Sebuah tracker dibutuhkan untuk mengkonsolidasi informasi tentang berkas dan peer dalam jaringan.
4. NodePeer terdiri dari 2 komponen utama yaitu Nodepeer Stream dan Nodepeer Serv.
5. Nodepeer Stream melakukan proses streaming berkas suara sedangkan Nodepeer Serv melakukan proses pelayanan streaming berkas suara.
6. Berdasarkan hasil pengujian tidak ditemukan penundaan yang mengganggu pemutaran berkas suara karena throughput jaringan lebih cepat dari pada bitrate berkas audio.
7. Berdasarkan hasil pengujian terdapat perbedaan pada berkas hasil unduhan dibandingkan berkas asli.
8. Berdasarkan hasil pengujian, beban terbagi hampir sama rata antar peer yang ada dalam sistem.

B. Saran

Saran yang dapat diberikan adalah :

1. Mengimplementasikan Error Checking sehingga dapat menghindari perbedaan berkas suara.
2. Menggunakan algoritma pelayanan antrian yang lebih baik.

DAFTAR REFERENSI

- [1] The World Bank . "Internet users as percentage of population." The World Bank Group. Web. 9 April 2013.
- [2] M. Hafeeda, Mohamed dan Barat K Bhargava. "On-Demand Media Streaming over the Internet".
- [3] Huang , Guowei dan Liang He. "Optimizing the Playback Quality of Data-Driven peer-to-peer Streaming."International Conference on Computational and Information Sciences. (2012).
- [4] Kao, Yung-Cheng dkk. "A Network Coding Equivalent Content Distribution Scheme for Efficient peer-to-peer Interactive VoD Streaming". IEEE Transactions On Parallel and Distributed System Vol. 23. (2012)
- [5] Tanenbaum, Andrew Stuart dan David J. Wetherall. Computer Networks, fifth edition. Perason Education ltd. (2011)
- [6] M. Hafeeda, Mohamed, Barat K Bhargava dan David K.Y. Yau. "A hybrid architecture for cost-effective on-demand media streaming". Computer Networks 44. (2004)