ANALISA PERFORMA SISTEM BERKAS EXT4 PADA KONDISI TERVIRTUALISASI

Salman Farizy[1], Tommy Gunawan[2]
[1,2] Program Studi Teknik Informatika, STMIK Eresha
JL. Raya Puspitek No 10, Buaran, Pamulang Tangerang Selatan

E-mail : sfarizy06@hotmail.com, kwee.tommygunawan@gmail.com

Abstrak --- *Beragamnya sistem berkas yang diciptakan serta digunakan oleh komputer dan sistem operasi maka jurnal ini dibuat untuk mempelajari lebih dalam salah satu sistem berkas yaitu ext4 dalam hal performanya untuk melakukan serangkaian tugas dari sebuah sistem berkas dengan menggunakan program tolak ukur khusus unix pada kondisi tervisualisasi dalam mesin virtual*

Kata Kunci--- *Ext4, Virtualisasi, Performa, Mesin Virtual, Sistem Berkas*

Abstract --- *Various file system that created and used by computer and operating system, so this journal is written for in depth learning about one of the file system called ext4 on how they complete various task of file system using a benchmarking program for unix in virtualized condition in virtual machine.*

Keyword--- *Ext4, Virtualization, Performance, Vitual Machine, File System*

1. PENDAHULUAN

Tidak dapat disanggah bahwa sebuah sistem berkas merupakan komponen paling penting dan berpengaruh dalam sebuah sistem operasi.

Hal ini dikarenakan sebuah sistem berkas bertugas sebagai pengatur dari berkas berkas dalam sebuah sistem operasi dan menyusunnya kedalam media menyimpanan yang sudah disediakan.

Selain dalam hal menyimpan dan mengatur berkas, sistem berkas juga berguna dalam memastikan data yang disimpan adalah data yang benar yang sesuai dengan pengguna maksud dengan menyimpan informasi dari berkas itu sendiri dalam bentuk *metadata.*

Dengan beragamnya sistem berkas yang beredar dan ditawarkan oleh sistem operasi berbasis unix terkadang timbul pertanyaan sistem berkas mana yang terbaik dan apa perbedaan dari setiap sistem berkas ini serta sejauh mana sistem berkas dapat berperforma dalam berbagai kondisi *hardware* dan pengaruh dari *hardware* itu terhadap performa sistem berkas oleh sebab itu penulis mencoba meneliti lebih lanjut sistem berkas ext4 dalam kondisi tervirtualisasi dalam mesin virtual.

2. TINJAUAN PUSTAKA

Dasar dari pembuatan jurnal ini mengacu pada salah satu jurnal teknologi yang disusun oleh Najib A. Kofahi, Ammar I. Al-Mubarak dan Ashraf Al-Jarrash dengan judul "***On Journaling File Systems in Linux: An Empirical Study***" yang membahas perbandingan performa dari beberapa sistem berkas pada sistem operasi linux ALT 2.2 yang akhirnya membuat penulis memutuskan untuk melakukan penelitian terhadap sistem berkas ext4 dalam kondisi tervirtualisasi dalam mesin virtual.

2.1. Konsep Sistem Berkas Ext4

Sistem berkas ext4 adalah sistem berkas hasil pengembangan dari sistem berkas yang umum digunakan sebelumnya yaitu ext3 dimana ext4 menawarkan berbagai perubahan fitur salah satunya adalah dengan menambahkan limitasi ukuran sistem berkas dari 16 *Terabyte* (TB) menjadi 1 *Exabyte* (EB) serta maksimal ukuran sebuah berkas menjadi 16 *Terrabyte* (TB).

Peningkatan lain yang ada dalam sistem berkas ext4 adalah menjadi tidak terhingga nya *sub directory* dari sebuah direktori yang sebelumnya dibatasi hanya sampai dengan 32000 *sub directory* saja.

selain itu fitur fitur lain yang ditawarkan adalah

- Multiblock allocation
- Delayed Allocation
- Fast File system consistency check (FSCK)
- dll.

2.2 Virtualisasi dan Mesin Virtual

Konsep dasar dari sebuah virtualisasi adalah membuat suatu bentuk maya dari sebuah sumber

daya yang ada sehingga sebuah sumber daya fisik dapat menjadi beberapa sumber daya yang akan digunakan sesuai kebutuhan.

Untuk saat ini hal yang paling umum dijadikan sebuah sumber daya virtualisasi adalah perangkat keras komputer, media penyimpanan dan sistem operasi.

Penulis menggunakan mesin virtual tipe II dimana mesin virtual ini membutuhkan sistem operasi pada *host* dari mesin virtual itu akan diletakan.

3. METODOLOGI PENELITIAN

Metode yang penulis gunakan dalam melakukan penelitian penulisan jurnal ini adalah dengan melakukan uji coba langsung oleh penulis dengan menggukanan bantuan batch program bernama bonnie++ buatan Russell Coker dengan versi 1.97.

3.1 Konfigurasi dan Spesifikasi Perangkat Keras Host

Dalam melakukan pengujian, host mesin virtual yang penulis gunakan adalah komputer dengan spesifikasi sebagai berikut:

- Prosesor Core i3 Gen4 3.40GHz
- RAM 8GB

Semua setting host dalam kondisi setingan awal tanpa dilakukan *overclocking* ataupun *tuning*

3.2 Konfigurasi dan Spesifikasi Mesin Virtual

Mesin virtual yang digunakan memiliki spesifikasi sebagai berikut:

- Prosesor satu core
- Ram 1GB
- Storage 20Gb
- Sistem operasi Ubuntu 16.04 LTS

Sebagai perbandingan untuk mengetahui sejauh mana sumber daya perangkat mempengaruhi kinerja dari sistem berkas maka penulis membuat mesin vitual baru yang hampir sama dengan spesifikasi sebagai berikut:

- Prosesor dua core
- Ram 1GB
- Storage 20Gb
- Sistem operasi Ubuntu 16.04 LTS

3.3 Metode, Alat, dan Parameter Pengujian

Metode pengujian yang penulis gunakan adalah pengujian tingkat kecepatan menggunakan program *batch* bonnie++ versi 1.97 sebanyak 3 perulangan secara langsung dengan kondisi host dalam keadaan *idle* tanpa terbuka applikasi apapaun yang mungkin dapat menggangu performa dari mesin virtual yang dijalankan serta ukuran yang digunakan adalah 2 *Gigabyte*.

Parameter yang digunakan dalam pengujian performa adalah sebagai berikut:

- Sequential Output
- Sequential Input
- Random Seek
- Sequential Create
- Random Create

selain itu, *latency* dari masing masing parameter pun di ukur untuk mendapatkan hasil yang lebih jelas seberapa lama jeda setiap proses.

4. HASIL DAN PEMBAHASAN

Setelah melakukan percobaan perulangan menggunakan bonnie++ sebanyak 3 perlulangan baik pada kondisi satu core dan dua core maka penulis mendapat data yang diolah menjadi beberapa bagian menurut parameternya.

4.1 Sequential Output

Tabel 1. Hasil Sequential Output Satu Core

Loop	Size	Sequential Output					
		Per Char		Block		Rewrite	
		K/sec	% CPU	K/sec	% CPU	K/sec	% CPU
1	2GB	1085	98	20150	2	5340	77
2	2GB	1035	97	15801	3	4125	76
3	2GB	1000	97	13928	1	3600	76

Pada tabel diatas bisa ditarik kesimpulan bahwa untuk menulis data secara *sequence* untuk ukuran sebesar 2 *Gigabyte* dibagi menjadi 3 yaitu per karakter, secara block dan ketika dilakukan penulisan ulang data.

Dalam perulangan pertama kecepatan penulisan per karakter secara *Sequence* mencapai 1085KB/sec dengan menggunakan sumber daya CPU sebanyak 98% dan terus mengalami penurunan pada loop ketiga sehingga jika dirata-ratakan kecepatan tulisnya adalah sebesar 1040KB/sec dengan rata rata penggunaan CPU sebesar 97,3%.

Untuk penulisan dalam *Block* pada perulangan pertama kecepatan nya 20150KB/sec dan hanya membebani cpu sebesar 2%, dan mengalami penurunan kecepatan seiring bertambahnya jumlah perulangan.

Pada penulisan ulang data kecepatan yang tercatat adalah 5340KB/sec dengan beban pada

CPU mencapai 77% dan seperti yang lain pada perulangan selanjutnya kecepatan tulis semakin menurun.

Sebagai perbandingan maka penulis melampirkan juga hasil Sequential Output ketika menggunakan mesin virtual dengan dua core.

Tabel 2. Hasil Sequential Output Dua Core

Loop	Size	Sequential Output					
		Per Char		Block		Rewrite	
		K /sec	% CPU	K /sec	% CPU	K /sec	% CPU
1	2GB	979	99	104895	21	28462	46
2	2GB	912	98	61303	9	13606	97
3	2GB	954	99	31338	7	6953	98

Pada penulisan per karakter didapat hasil yang cukup menarik dimana terjadi penurunan kecepatan sekitar 9.7% dari yang satu core.

Namun peningkatan signifikan terjadi pada penulisan block sampai lima kali lipat dari yang satu *core* pada perulangan pertama.

Sebagai data tambahan penulis juga melampirkan *latency* dari masing masing proses dari kedua mesin virtual

Tabel 3. Hasil latency Sequential Output Satu Core

Loop	Size	Sequential Output		
		Per Char	Block	Rewrite
		Latency	Latency	Latency
1	2GB	13736us	1166ms	894ms
2	2GB	13581us	1410ms	1532ms
3	2GB	29036us	1566ms	1494ms

Tabel 4. Hasil latency Sequential Output Dua Core

Loop	Size	Sequential Output		
		Per Char	Block	Rewrite
		Latency	Latency	Latency
1	2GB	22758us	454ms	454ms
2	2GB	40184us	361ms	500ms
3	2GB	25322us	1149ms	795ms

Latency disini adalah jeda yang tercatat sebelum setiap perintah dijalankan oleh mesin virtual ukuran yang digunakan adalah *microsecond*(us) dan *milisecond*(ms).

4.2 Sequential Input

Tabel 5. Hasil Sequential Input Satu Core

Loop	Size	Sequential Input			
		Per Char		Block	
		K/sec	% CPU	K/sec	% CPU
1	2GB	4283	97	16723	92
2	2GB	3545	97	10621	95
3	2GB	3894	96	8925	98

Sequential input adalah kecepatan baca sebuah data secara *sequence* atau berurutan. dari hasil ujicoba didapatkan data seperti pada tabel diatas, pada segmentasi per karakter pada perulangan pertama kecepatan yang dicapai adalah sebesar 4283KB/sec dengan menggunakan CPU sebesar 97%.

Pada segmentasi pembacaan block kecepatan yang tercatat adalah sebesar 16723KB/sec dengan pemakaian CPU sebesar 92%.

Selama perulangan pada bagian pembacaan per karakter terjadi penaikan kecepatan pada perulangan terakhir namun pada block setiap perulangan mengalami penurunan kecepatan.

Tabel 6. Hasil Sequential Input Dua Core

Loop	Size	Sequential Input			
		Per Char		Block	
		K/sec	% CPU	K/sec	% CPU
1	2GB	630	97	48374	74
2	2GB	4072	97	26797	77
3	2GB	2932	90	22178	93

Hal yang sama terjadi pada pembacaan data secara sequential per karakter dimana penurunan kecepatan terjadi kembali. namun penurunan ini terjadi tidak sesignifikan yang terjadi pada penulisan.

Untuk data latency bisa dilihat pada kedua tabel dibawah ini.

Tabel 7. Hasil latency Sequential Input Satu Core

Loop	Size	Sequential Input	
		Per Char	Block
		Latency	Latency
1	2GB	17393us	120ms
2	2GB	16150us	115ms
3	2GB	22784us	58113us

Tabel 8. Hasil latency Sequential Input Dua Core

Loop	Size	Sequential Input	
		Per Char	Block
		Latency	Latency
1	2GB	17953us	79771us
2	2GB	19268us	56689us
3	2GB	66105us	70404us

4.3 Random Seek

Random seek disini mewakili aspek yang penting dari sebuah sistem berkas yaitu IOPS dimana IOPS ini adalah representasi dari kecepatan baca dan tulis secara random maka kecepatan yang dipakai adalah banyaknya aksi baca tulis per detik.

Dari 3 perulangan baik untuk mesin virtual satu core dan dua core didapat data seperti tabel berikut

Tabel 9. Hasil Random Seek Satu Core

Loop	Size	Random Seeks	
		/sec	% CPU
1	2GB	179	68
2	2GB	208	98
3	2GB	187	111

Tabel 10. Hasil Random Seek Dua Core

Loop	Size	Random Seeks	
		/sec	% CPU
1	2GB	199.1	46
2	2GB	190.4	59
3	2GB	183.4	79

Dari tabel di atas, bisa dilihat bahwa jumlah IOPS dari mesin virtual satu core mengalami fluktuatif disetiap perulangan dengan peningkatan penggunaan CPU di setiap perulangannya. namun pada mesin virtual dua core penurunan terjadi namun tidak terlalu sigkinifikan dengan peningkatan CPU yang tidak terlalu drastis juga.

untuk latency sendiri ternyata perbedaan untuk satu core dan dua core tidak terlalu signifikan pada saat menjalankan test ini.

Tabel 11. Hasil latency Random Seeks Satu Core

Loop	Size	Random Seeks
		Latency
1	2GB	1093ms
2	2GB	1136ms
3	2GB	980ms

Tabel 12. Hasil latency Random Seek Dua Core

Loop	Size	Random Seeks
		Latency
1	2GB	1092ms
2	2GB	526ms
3	2GB	1147ms

4.4 Sequential Create dan Random Create

Sequential create pada pengujian ini mewakili banyaknya file data yang dibuat, dibaca dan dihapus perdetik baik secara *sequence* maupun secara *random*.

dari pengujian dua mesin virtual yang ada data beberapa data tidak tercatat atau terdefinisikan dengan sempurna dikarenakan proses yang selesai lebih cepat dari yang bisa dideteksi oleh program bonnie++ dan data itu direpresentasikan dengan simbil positif (+). Berikut adalah penjabaran data tersebut dalam bentuk tabel:

Tabel 13. Hasil Sequential Create Satu Core

Loop	Num Files	Sequential Create					
		Create		Read		Delete	
		/sec	% CPU	/sec	% CPU	/sec	% CPU
1	16	12511	54	+++++	+++	21746	51
2	16	18156	54	+++++	+++	23705	54
3	16	16658	56	+++++	+++	20128	53

Tabel 14. Hasil Sequential Create Dua Core

Loop	Num Files	Sequential Create					
		Create		Read		Delete	
		/sec	% CPU	/sec	% CPU	/sec	% CPU
1	16	+++++	+++	+++++	+++	+++++	+++
2	16	14499	81	+++++	+++	+++++	+++
3	16	12989	75	+++++	+++	27272	55

Tabel 15.Hasil Random Create Satu Core

Loop	Num Files	Random Create					
		Create		Read		Delete	
		/sec	% CPU	/sec	% CPU	/sec	% CPU
1	16	20329	62	+++++	+++	23656	53
2	16	21051	54	+++++	+++	23663	54
3	16	17816	53	+++++	+++	20143	54

Tabel 16. Hasil Random Create Dua Core

Loop	Num Files	Random Create					
		Create		Read		Delete	
		/sec	% CPU	/sec	% CPU	/sec	% CPU
1	16	+++++	+++	++++ +	+++	+++++	+++
2	16	+++++	+++	++++ +	+++	+++++	+++
3	16	22959	56	++++ +	+++	25840	56

5. SIMPULAN

Setelah meneliti dan mendalami serta mengolah seluruh data hasil uji coba penulis dapat menyimpulkan bahwa sistem berkas ext4 bekerja sangat baik walaupun dalam kondisi tervisualisasi dalam mesin virtual yang disetting dengan spesifikasi rendah.

Walau dalam beberapa aspek perbedaan spesifikasi terlihat tidak mengalami perubahan namun jika dilihat secara keseluruhan maka korelasi dari performa ext4 dengan spesifikasi maka peningkatan yang terjadi sangat signifikan sehingga tidak heran jika sistem berkas ext4 dapat memanfaatkan setiap sumber daya yang ada dengan baik.

6. SARAN

Bonnie++ adalah program pengecekan performa sistem berkas yang baik, namun bukan satu satunya yang ada diluar sana, maka penulis mengharapkan bahwa jika ada yang melakukan penelitian sejenis menggunakan sistem berkas yang lain bisa menggunakan program *benchmarking* untuk unix yang lain agar bisa menjadi acuan baru.

Selain itu pengujian sistem berkas bukan hanya bisa dilakukan dengan bantuan program manun juga dapat dengan pengujian secara *real time* sesuai dengan kondisi nyata ketika sistem berkas ini dipakai secara sehari hari dan hal tersebut bisa menjadi materi penelitian lanjutan.

DAFTAR PUSTAKA

[1] Najib A. Kofahi, Ammar I. Al-Mubarak and Ashraf Al-Jarrash.(2012). On Journaling File Systems in Linux : An Empirical Study, https://www.semanticscholar.org/paper/On-Journaling-File-Systems-in-Linux-%3A-An-Empirical-Kofahi-Al-Mubarak/e65b1f9dbfadb75da91b20895e658369e3c58808, diakses terakhir pada 09 Januari 2020