OPTIMASI PEMROSESAN ENKRIPSI DAN DEKRIPSI RSA PADA SINGLE BOARD COMPUTER (SBC) DENGAN PEMBAGIAN BEBAN KOMPUTASI DALAM SISTEM TERDISTRIBUSI

Arief Prasetyo[1], Sofyan Noor Arief[2], Rokhimatul Wakhidah[3]

[1,2,3] Teknik Informatika, Teknologi Informasi, Politeknik Negeri Malang
[1] arief.prasetyo@polinema.ac.id, [2] sofyan@polinema.ac.id, [3] wakhidah@polinema.ac.id

Abstrak

Algoritma kriptografi RSA menjadi metode yang paling sulit diretas karena kerumitan persamaan matematis yang digunakan. Semakin rumit persamaan matematis yang digunakan, semakin besar pula daya komputasi yang digunakan untuk memproses algoritma tersebut. Penggunaan Single-Board Computer (SBC) mulai digunakan untuk menggantikan alat pemrosesan komputasi seperti komputer personal baik berupa laptop maupun workstation. Penetlitian ini bertujuan untuk mengetahui hasil optimasi pemroresan enkirpsi dan dekripsi RSA menggunakan SBC yang tersusun sebagai bentuk cluster. Metode yang digunakan adalah metode pemrosesan enkripsi dan dekripsi RSA dengan pembagian beban komputasi yang diterapkan dalam arsitektur SBC berdasarkan ketersediaan sumberdaya pada tiap SBC dalam *cluster* SBC. Cluster SBC yang dibangun terdiri atas 5 buah SBC Raspberry Pi 4 model B yang terkoneksi melalui sebuah switch 8 port. Hasil penelitian yang didapat adalah proses enkripsi dan dekripsi pada cluster SBC dapat mengalahkan komputer tunggal. Cluster SBC yang memiliki 16 core dari 4 buah prosesor worker SBC berjalan lebih cepat dibandingkan komputer tunggal yang memiliki 4 core dari prosesornya. Selain itu, efisiensi juga didapat dari biaya yang dikeluarkan untuk membeli perangkat. Dengan biaya pembelian yang sama, performa yang didapat lebih tinggi pada cluster SBC sehingga SBC dapat dijadikan alternatif pemrosesan enkripsi dan dekripsi.

Kata kunci : RSA, SBC, cluster SBC

1. Pendahuluan

Algoritma kriptografi RSA telah banyak diimplementasikan untuk pengamanan file teks (Manurung, Sirait, & Panggabean, 2018) (Pahrizal & Pratama, 2016), database (Wibowo, Susanto, & Shamir, 2009), email (Ginting, Isnanto, & Windasari, 2015). Algoritma kriptografi RSA menjadi metode paling populer digunakan karena metode ini cukup susah untuk diretas. Kesulitan dalam meretas metode ini dikarenakan kerumitan persamaan matematis yang digunakan dalam metode tersebut. Semakin rumit persamaan matematis yang dipakai maka semakin besar pula sumber daya komputasi yang harus digunakan untuk memprosesnya. Kekuatan modular pada algoritma enkripsi RSA memiliki andil besar dalam mempengaruhi kinerja algoritma (Wandert, Gura, Eberle, Gupta, & Shantz, 2005) dan menjadi permasalahan utama yang menjadi hambatan dalam penerapan algoritma pada sebuah aplikasi yang lebih kompleks.

Permasalahan muncul ketika sumber daya komputasi yang dibutuhkan dalam proses enkripsi dan dekripsi dengan algoritma RSA melebihi kemampuan sumber daya tunggal yang tersedia. Penambahan sumber daya komputasi tidak serta merta menyelesaikan masalah, karena diperlukan kolaborasi dan komunikasi antar sumber daya pemrosesan agar dapat berjalan maksimal. Dengan teknologi *computer clustering* di mana beberapa perangkat akan digunakan secara bersama-sama dengan membagi sumber daya secara terdistribusi menjadi jawaban yang tepat untuk mengatasi permasalahan tersebut. Dengan sistem terdistribusi, sekelompok perangkat pemrosesan/komputer akan melakukan kolaborasi dan komunikasi untuk memproses pekerjaan yang diberikan kepada masing-masing pekerjanya. Dengan menggunakan sistem terdistribusi, waktu pemrosesan akan jauh lebih cepat dibandingkan dengan komputasi tunggal yang tidak terdistribusi. Dengan menggunakan sistem terdistribusi, waktu pemrosesan akan jauh lebih cepat dibandingkan dengan komputasi tunggal yang tidak terdistribusi (Lee & Lee, 2009).

Semakin majunya ilmu pengetahuan dan teknologi saat ini membuat perkembangan alat pemrosesan komputasi menjadi semakin berkembang. Penggunaan Single Board Computer (SBC) pun mulai sering digunakan untuk menggantikan alat pemrosesan komputasi yang umumnya adalah sebuah komputer personal baik berupa laptop maupun workstation.

Penelitian implementasi SBC Raspberry Pi 3 Model B+ dalam kasus pendeteksian gulma pada lahan pertanian mampu mengungguli performa

komputer personal (berbasis x86 intel core i5) sebanyak 0,04 kali lebih cepat. Selain itu penggunaan SBC pada kasus tersebut dapat lebih menghemat biaya karena SBC yang dipakai berharga jauh dibawah komputer personal pembandingnya (Sukoco, Solahudin, & Suriansyah, 2015).

Penelitian yang mengimplementasikan arsitektur *cluster* SBC dalam kasus pemrosesan *machine-learning*. Dengan mengimplementasikan arsitektur tersebut, performa pemrosesan meningkat secara signifikan jika dibandingkan dengan pemrosesan pada SBC tunggal (Irvi, 2019). Pemrosesan tunggal sangat terbatas oleh sumber daya yang dimilikinya. Namun pemrosesan secara parallel pada *cluster* SBC tidak menunjukkan peningkatan performa yang signifikan ketika pekerjaan yang dikerjakan tidak terlalu berat. Karena pemecahan dan komunikasi dalam pembagian porsi pekerjaan membutuhkan waktu tambahan.

Penelitian ini bertujuan untuk mengetahui optimasi pemrosesan enkripsi dan dekripsi RSA dengan pembagian beban komputasi yang diterapkan dalam arsitektur cluster SBC. Metode yang diusulkan yaitu membagi beban komputasi enkripsi dan dekripsi RSA berdasarkan ketersediaan sumberdaya pada masing-masing SBC dalam cluster SBC. Sumber daya yang dijadikan parameter dalam perhitungan beban yang dikerjakan meliputi sumber daya pemrosesan CPU dan Memory.

Algoritma Kriptografi RSA merupakan salah satu algoritma kriptografi kunci publik. Algoritma ini adalah algoritma pertama yang cocok dalam melakukan proses enkripsi dan dekripsi dengan menggunakan metode asimetrik.

RSA terbagi menjadi tiga proses, yaitu pembangkitan kunci, enkripsi dan dekripsi. Dasar proses enkripsi dan dekripsi pada algoritma RSA yaitu konsep bilangan prima dan aritmatika modulo. Kunci enkripsi tidak dirahasiakan dan diberikan kepada umum (disebut kunci publik), sedangkan kunci untuk dekripsi bersifat rahasia (disebut kunci pribadi). Untuk menemukan kunci dekripsi, dilakukan dengan cara memfaktorkan bilangan bulat menjadi faktor-faktor primanya.

Keamanan algoritma RSA terletak pada sulitnya memfaktorkan bilangan yang besar menjadi faktor-faktor prima. Pemfaktoran dilakukan untuk memperoleh kunci privat. Selama pemfaktoran bilangan besar menjadi faktor-faktor prima belum ditemukan algoritma yang mangkus, maka selama itu pula keamanan algoritma RSA tetap terjamin (Munir, 2008).

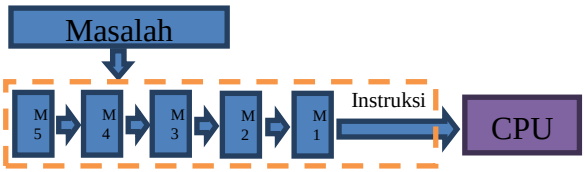
Single-board computer (SBC) adalah komputer yang lengkap dibangun pada papan sirkuit tunggal, berikut mikroprosesor(s), memori, input / output (I/O) dan fitur lain yang dibutuhkan pada sebuah komputer fungsional. Komputer single-board dibuat termasuk sebagai platform pengembangan sistem, untuk sistem pendidikan, atau untuk digunakan sebagai pengendali komputer tertanam (*embedded*). Banyak jenis komputer portabel yang mengintegrasikan semua fungsi mereka ke sebuah papan sirkuit tunggal. Contoh SBC dapat dilihat pada Gambar 1.

Gambar 1. Contoh SBC

Raspberry Pi, merupakan salah satu produk komputer papan tunggal (single-board circuit; SBC) yang seukuran dengan kartu kredit yang dapat digunakan untuk menjalankan program perkantoran, permainan komputer, dan sebagai pemutar media hingga video beresolusi tinggi. Raspberry Pi dikembangkan oleh yayasan nirlaba, Rasberry Pi Foundation, yang digawangi sejumlah pengembang dan ahli komputer dari Universitas Cambridge, Inggris.

Komputasi terdistribusi adalah bidang ilmu komputer yang mempelajari sistem terdistribusi. Sebuah sistem terdistribusi terdiri dari beberapa komputer otonom yang berkomunikasi melalui jaringan komputer. Komputer berinteraksi satu sama lain untuk mencapai tujuan bersama. Sebuah program komputer yang berjalan dalam sistem terdistribusi disebut program yang didistribusikan, dan pemrograman terdistribusi adalah proses penulisan program tersebut (Andrews, 1999)

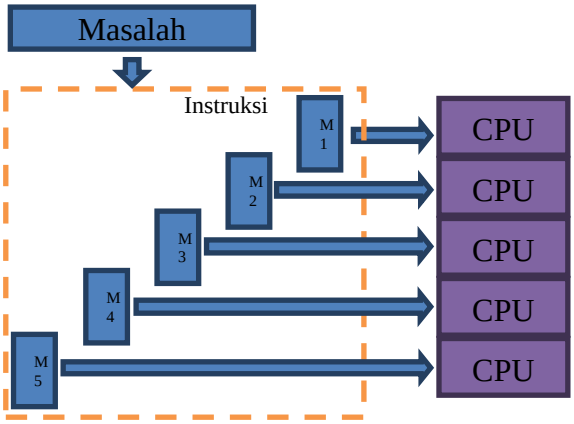
Cara kerja komputasi terdistribusi (Andrews, 1999) adalah beban kerja akan didistribusikan ke komputer-komputer yang terhubung untuk diselesaikan di mana semua itu dikendalikan oleh suatu sistem operasi. Pengguna hanya cukup melakukan pengaturan sistem operasi, kemudian sistem operasilah yang akan melakukan tugasnya mengorganisasi kemampuan dan tugas ke komputer-komputer itu.



Gambar 2. Komputasi Pada Sistem Komputer Tunggal

Ciri khas dari komputasi terdistribusi (Andrews, 1999) adalah heterogenitas dalam berbagai hal seperti perangkat keras, sistem operasi, dan bahasa pemrograman karena tidak mungkin untuk mengembangkan sistem terdistribusi yang homogen, karena secara alamiah sistem komputer terdistribusi tumbuh dari lingkungan yang heterogen. Kata kunci dalam menjembatani perbedaan-perbedaan yang muncul adalah

interoperabilitas (*interoperability*). Ciri lain dari komputasi terdistribusi adalah di mana pemakai tidak perlu menyadari komputer mana yang bekerja untuk melaksanakan tugas komputasi. Ibaratnya, pemakai ingin ini dan mendapat hasil komputasi yang diinginkan tanpa memandang oleh siapa pekerjaan itu dikerjakan. Semua alokasi sumber daya dan penanganan kerja dikendalikan oleh sistem operasi. Dicirikan pula menggunakan banyak komputer yang saling terhubung dalam suatu jaringan komputer, untuk melakukan komunikasi proses antar komputer yang bekerja.



Gambar 3. Komputasi Pada Sistem Komputer Terdistribusi

2. Metodologi Penelitian

2.1 Teknik Pengumpulan Data

Ada dua teknik pengumpulan data, yaitu data primer dan data sekunder.

1) Data primer

Data didapatkan dari hasil membagi porsi pekerjaan pemrosesan enkripsi dan dekripsi kepada seluruh *worker* yang tersedia dalam *cluster SBC* tersebut secara merata. Penelitian ini hanya mencakup proses enkripsi dan deksripsi-nya saja, dimana proses generasi kunci *public* dan *private* tidak termasuk.

2) Data sekunder

Data diperoleh dengan cara studi literatur dengan mendalami penelitian dari buku dan jurnal yang terkait. dengan enkripsi dan dekripsi RSA, Single Board Computer (SBC) Raspberry Pi 4 Model B, *cluster* SBC dan distribusi beban komputasi dalam sistem terdistribusi.

2.2 Perancangan Metode dan Algoritma

1) Pembagian/ Pemecahan file

Proses pertama dimulai dengan adanya pembagian file yang akan dikerjakan menjadi bagian-bagian kecil sesuai dengan block dari kunci enkripsinya. Pembagian *file* tersebut menggunakan utilitas standar yang telah ada di sistem operasi *unix* yaitu perintah *split* (Kerrisk, 2020a). Adapun keuntungan dari menggunakan utilitas tersebut adalah sudah tersedianya utilitas tersebut di semua sistem operasi *unix* yang biasa digunakan oleh *Single Board Computer (SBC)* dan kemampuannya untuk memecah *file* tanpa menambahkan mekanisme tambahan apapun karena *file* yang dipecah akan dibaca sebagai *RAW file*.

Utilitas tersebut dapat digunakan dengan menjalankan perintah *split <nama_file_yang_akan_dipecah>* pada terminal *unix* dan pastikan anda berada dalam direktori dari *file* yang akan dipecah. Dalam penelitian ini, perintah *split* yang akan digunakan adalah perintah *split* dengan opsi besaran *file* hasil dari proses pemecahan tersebut. Detil perintah yang harus dijalankan yaitu *split -b <block_size> <nama_file_yang_akan_dipecah>*.

2) Penyebaran file hasil proses pemecahan

Setelah proses pemecahan file, proses berikutnya adalah penyebaran file hasil proses pemecahan kepada masing-masing pekerja pada cluster SBC. Pada proses ini, membuat file sharing antar anggota dalam cluster SBC memanfaatkan protokol Network File System yang telah disediakan oleh sistem operasi *unix* sebagai standar *file sharing* dalam sistem operasinya (Sun Microsystem, 1989). Keunggulan dari protokol ini adalah kemudahan dalam konfigurasi dan performanya yang bagus. Dengan memanfaatkan protokol tersebut, *file* hasil pemecahan akan dibagi dengan cara memasang (*mount*) direktori *file sharing* yang berisi *file* yang telah dipecah tersebut ke masing-masing anggota *cluster SBC*. Sehingga semua anggota dalam *cluster SBC* dapat mengakses *file* tersebut.

3) Proses enkripsi pecahan-pecahan file

Setelah proses penyebaran dari pecahan *file* yang akan dienkripsi, langkah berikutnya adalah memproses pecahan-pecahan *file* tersebut dalam masing-masing komputer pekerja yang ada dalam *cluster SBC*. Pada proses ini, dibuatlah sebuah aplikasi *python* sederhana dengan memanfaatkan *library Crypto* yang telah tersedia di bahasa pemrograman *python* (Python Software Foundation, 2020a). Masing-masing pekerja akan dikontrol oleh broker melalui komunikasi *socket* yang dibuat berdasarkan utilitas *XMLRPC* yang terdapat dalam bahasa pemrograman *python* (Python Software Foundation, 2020b). Kunci yang digunakan untuk proses enkripsi dan dekripsi juga akan diberikan oleh broker dan akan disebar juga menggunakan sistem *sharing* seperti proses penyebaran *file* pada langkah sebelumnya.

4) Penyatuan file hasil enkripsi

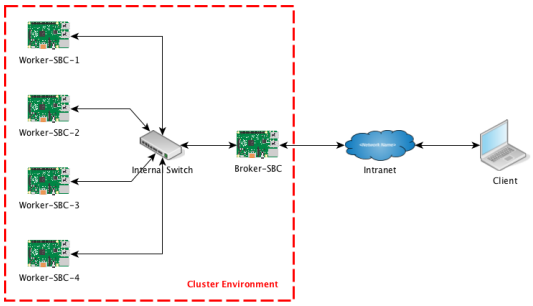
Setelah proses enkripsi terjadi di masing-masing komputer pekerja dalam sistem *cluster SBC*, langkah berikutnya adalah menyatukan *file* hasil enkripsi sesuai dengan urutan pemecahan *file*-nya. Pada proses ini, kembali digunakan utilitas yang telah disediakan pada sistem operasi *unix* yaitu utilitas *cat* (Kerrisk, 2020b). Utilitas ini dapat dijalankan melalui terminal dengan menjalankan

perintah *cat <file-1> <file-2>…. <file-x> > <nama_file_hasil>*. Hasil dari proses penggabungan *file* yang telah terenkripsi akan disimpan ke dalam sebuah direktori tersendiri, dan di dalamnya akan disertakan *key* dalam bentuk *file* yang digunakan untuk proses dekripsinya serta keterangan *padding* yang ditambahkan dalam proses enkripsi tersebut.

5) Proses dekripsi file

Untuk proses dekripsi *file*, proses akan mulai kembali dari langkah awal yaitu pemecahan *file*. Namun untuk pemecahan *file* pada proses dekripsi, *file* tidak dipecah berdasarkan ukuran *block* saja melainkan berdasarkan ukuran *block* dan *padding* yang ditambahkan dari proses enkripsi. Kemudian proses akan berjalan sama dengan proses enkripsi, hingga *file* hasil dekripsi selesai diproses dan disimpan dalam direktori tersendiri.

2.3 Perancangan *Testbed*



Gambar 4. Desain Arsitektur Testbed

Metode yang diajukan dalam penelitian ini akan diujikan dalam lingkungan sistem yang sesungguhnya. Adapun detil lingkungan pengujian/testbed-nya dapat digambarkan pada Gambar 4.

2.4 Pengujian

Pengujian dilakukan untuk membuktikan bahwa penerapan arsitektur pembagian beban dapat berjalan dengan baik pada cluster Single Board Computer (SBC). Selain itu pengujian juga dilakukan untuk mengukur sejauh mana performa cluster SBC dapat menyelesaikan kasus enkripsi yang diberikan kepadanya. Pengujian akan dilakukan dengan melakukan proses enkripsi dan dekripsi pada 10 buah berkas dokumen dengan ukuran yang berbeda-beda. Dokument tersebut masing-masing berukuran 10MB, 20MB, 30MB, 40MB, 50MB, 60MB, 70MB, 80MB, 90MB, dan 100MB. Proses pengujian akan dilakukan pada dua lingkungan uji yang berbeda yaitu pada cluster SBC dan pada komputer tunggal multicore.

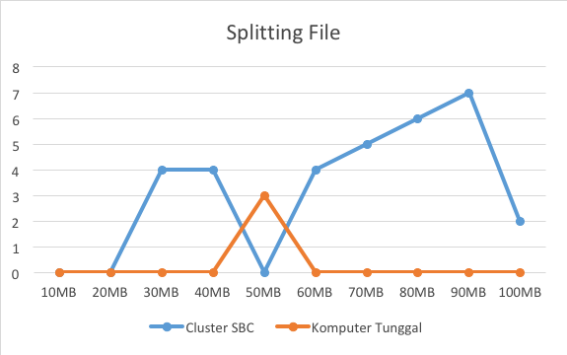
Komputer tunggal yang akan digunakan paada proses ujicoba merupakan komputer tunggal dengan harga yang sama dengan total seluruh perangkat yang digunakan pada cluster SBC. Hal ini dimaksudkan untuk membandingkan efisiensi penggunaan cluster SBC dengan komputer tunggal dalam kasus menyelesaikan pekerjaan ekripsi dan dekripsi. Komputer tunggal yang akan digunakan pada pengujian ini adalah komputer dengan dengan jumlah core yaitu 4 core. Komputer tunggal tersebut berjalan pada frekuensi 2,1 Ghz.

SBC yang digunakan dalam lingkungan pengujian ini merupakan Raspberry Pi 4 Model B sebanyak 5 buah yang masing-masing memiliki spesifikasi sebagai berikut:

- *Processor* : Broadcom BCM2711, Quad core Cortex-A72 SoC @ 1.5GHz
- *RAM* : 4GB LPDDR4-3200 SDRAM
- *Storage* : 32 GB
- *NIC* : 10/100/1000 Mbps Ethernet
- Sistem Operasi : Raspbian

3. Hasil Pengujian dan Diskusi

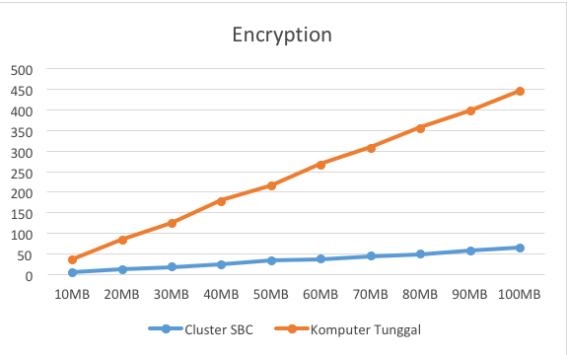
3.1 Pembagian/ pemecahan file



Gambar 5. Perbandingan waktu pemecahan file

Pada Gambar 5, proses pemecahan file cluster SBC membutuhkan waktu lebih banyak dibandingkan dengan pada proses tunggal. Hal ini terjadi karena clock speed dari prosesor pada komputer tunggal lebih tinggi dibandingkan dengan clock speed pada arsitektur cluster SBC.

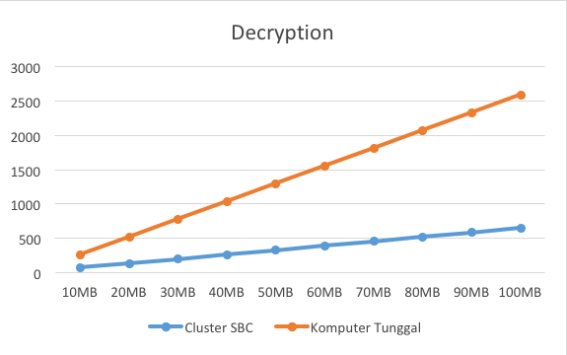
3.2 Proses enkripsi pecahan-pecahan file



Gambar 6. Proses enkripsi pecahan-pecahan file

Pada proses enkripsi, cluster SBC dapat mengalahkan komputer tunggal seperti yang terlihat pada Gambar 6. Proses enkripsi dapat terselesaikan

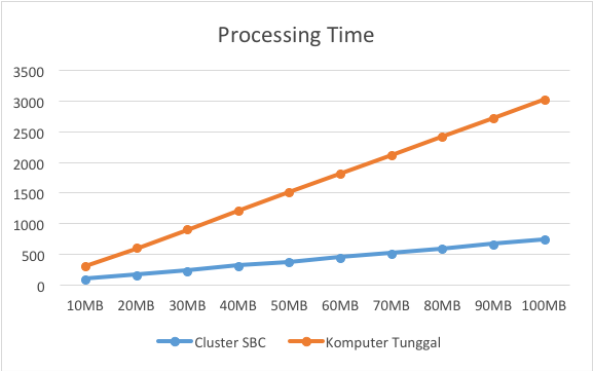
lebih cepat pada cluster SBC. Hal ini dikarenakan semua core yang tersedia akan digunakan seluruhnya pada proses enkripsi. Semakin banyak jumlah core yang dipunyai, prosesnya akan berjalan lebih cepat. Cluster SBC yang memiliki total core sejumlah 16 core yang berasal dari 4 buah processor pada worker SBC, sedangkan komputer tunggal hanya memiliki total 4 core yang berasal dari 1 buah processor-nya.



Gambar 7. Proses dekripsi pecahan-pecahan file

Pada proses dekripsi, cluster SBC juga memiliki waktu pemrosesan lebih cepat dari komputer tunggal seperti pada Gambar 7. Semakin besar berkas file, terlihat perbedaan waktu pemrosesan juga semakin lama.

Perolehan waktu pemrosesan waktu total pun menjadi lebih cepat pada cluster SBC seperti pada Gambar 8. Hal ini terjadi karena inti dalam pemrosesan enkripsi dan dekripsi yaitu tahapan proses enkripsi dan dekripsi berjalan lebih cepat pada cluster SBC. Sehingga dapat ditarik kesimpulan bahwa penggunaan arsitektur cluster SBC tepat digunakan untuk pemrosesan ekripsi dan dekripsi berkas karena dapat menurunkan waktu pemrosesan secara signifikan.



Gambar 8. Waktu Pemrosesan

4. Kesimpulan dan Saran

4.1 Kesimpulan

Berdasarkan percobaan yang telah dilakuan, dapat ditarik kesimpulan bahwa:

a) Proses enkripsi dan dekripsi dapat terselesaikan lebih cepat pada *cluster* SBC. Hal ini dikarenakan semua *core* yang tersedia akan digunakan seluruhnya pada proses enkripsi dan dekripsi. Sehingga, semakin banyak jumlah *core* yang dipunyai, prosesnya akan berjalan lebih cepat. *Cluster* SBC yang memiliki total *core* sejumlah 16 *core* yang berasal dari 4 buah *processor* pada *worker* SBC, sedangkan komputer tunggal hanya memiliki total 4 *core* yang berasal dari 1 buah *processor*-nya.

b) Perolehan total waktu pemrosesan pun menjadi lebih cepat pada *cluster* SBC. Hal ini terjadi karena inti dalam pemrosesan enkripsi dan dekripsi yaitu tahapan proses enkripsi dan dekripsi berjalan lebih cepat pada *cluster* SBC.

c) Penggunaan arsitektur *cluster* SBC tepat digunakan untuk pemrosesan ekripsi dan dekripsi berkas karena dapat menurunkan waktu pemrosesan secara signifikan dengan biaya yang lebih murah.

4.2 Saran

Berdasarkan penelitian yang telah dilakukan, terdapat beberapa saran yang dapat dilakukan dalam penelitian selanjutnya. Saran tersebut antara lain:

a) Pembuatan prototipe yang lebih user friendly agar pengguna dapat memasukkan parameter-parameter percobaan tanpa harus mengubah secara langsung kode program

b) Penggunaan algoritma enkripsi dan dekripsi lain pada desain pemrosesan yang diajukan dalam penelitian ini

c) Pembuatan produk jadi penyimpanan file terenkripsi yang mengimplementasikan metode yang telah dikembangkan dalam penelitian ini

Daftar Pustaka:

Andrews, G. R. (1999). *Foundations of Multithreaded, Parallel, and Distributed Programming*. Addison-Wesley.

Ginting, A., Isnanto, R. R., & Windasari, I. P. (2015). Implementasi Algoritma Kriptografi RSA untuk Enkripsi dan Dekripsi Email. *Jurnal Teknologi Dan Sistem Komputer*, *3*(2), 253–258.

Irvi, E. (2019). Rancang Bangun dan Evaluasi Kinerja Raspberry Pi Cluster sebagai Platform Penerapan Pembelajaran Mesin.

Kerrisk, M. (2020a). *Cat Unix Manual*.

Kerrisk, M. (2020b). *Split Unix Manual*.

Lee, S., & Lee, E. (2009). Distributed adaptation system for quality assurance of web service in mobile environment. In *Journal of Information Science and Engineering* (Vol. 25).

Manurung, J., Sirait, K., & Panggabean, J. F. (2018). Penerapan Algoritma Rsa Untuk Pengamanan

File. *Jurnal Mantik Penusa*, *2*(2), 112–116.

Munir, R. (2008). Pengantar Ilmu Kriptografi. *Penerbit Andi*.

Pahrizal, P., & Pratama, D. (2016). Implementasi Algoritma Rsa Untuk Pengamanan Data Berbentuk Teks. *Pseudocode*, *3*(1), 44–49. https://doi.org/10.33369/pseudocode.3.1.44-49

Python Software Foundation. (2020a). *Cryptographic Services*.

Python Software Foundation. (2020b). *xmlrpc — XMLRPC server and client modules*.

Sukoco, H., Solahudin, M., & Suriansyah, M. I. (2015). Perbandingan Kinerja Pemrosesan Paralel Pada PC dan Raspberry Pi Untuk Pendeteksian Gulma Pada Lahan Pertanian Menggunakan Fraktal.

Sun Microsystem, I. (1989). *NFS: Network File System Protocol Specification*.

Wandert, A. S., Gura, N., Eberle, H., Gupta, V., & Shantz, S. C. (2005). Energy analysis of public-key cryptography for wireless sensor networks. In *Proceedings - Third IEEE International Conference on Pervasive Computing and Communications, PerCom 2005* (Vol. 2005). https://doi.org/10.1109/percom.2005.18

Wibowo, I., Susanto, B., & Shamir, A. (2009). Penerapan algoritma kriptografi asimetris rsa untuk keamanan data di oracle. *Jurnal Informatika*, *5*(1).