Jurnal Sains Komputer & Informatika (J-SAKTI)
Volume 6 Nomor 1, Maret 2022, pp. 126-135
ISSN: 2548-9771/EISSN: 2549-7200
https://tunasbangsa.ac.id/ejurnal/index.php/jsakti

Document Management System Menggunakan Kombinasi Algoritma ROT13 dan Algoritma String Matching

Feri Setiawan[1], Fauziah[2]
[1,2]Informatika, Fakultas Teknologi komunikasi dan Informatika, Universitas Nasional
ferisetiawan239@gmail.com, fauziah@civitas.unas.ac.id

Abstract

Technology for document management is really need because it can improve performance both in storage, search and document security. Document management in a manual way can waste time in finding required documents and requiring a lot of space for hardcopy document storage. This study uses a string matching algorithm combined with ROT13 algorithm. String matching algorithm is used to find a file name in database, the file name in database has been encrypted for example plaintext "Kabel Ladder" into chipertext "Xnory Ynqqre". The steps of string matching algorithm combined with ROT13 algorithm in this study are, word pattern is encrypted and then matched with existing data in database and then a string search in performed using string matching algorithm and produces the data sought. In both algorithm, time and number of matches were tested between pattern and data to show performance of two algorithms using 200 data until 500 data. The result of testing 200 data for search word Tower takes 0.00513 sec and matches 65x, 300 data for search word Tower takes 0.00716 sec and matches 164x, 400 data for search word Tower takes 0.01372 sec and matches 262x and 500 data for search word Tower takes 0.02233 sec and matches 360x. From the test result in this study, the researchers concluded that string matching algorithm combined with ROT13 algoritm, the more data, more time it will take, even though it is still relatively small.

Keywords*: Document, DMS, ROT13 Algorthm, String Matching Algorithm*

Abstrak

Pengelolaan dokumen dengan cara manual dapat membuang waktu dalam pencarian dokumen yang dibutuhkan membutuhkan dan banyak tempat untuk penyimpanan hardcopy dokumen. Penelitian ini menggunakan algoritma string matching yang dikombinasikan dengan algoritma ROT13. Algoritma string matching digunakan untuk mencari sebuah nama file yang terdapat pada database, nama file didatabase telah di enskripsi dengan algoritma ROT13 contoh plaintext "Kabel Ladder" menjadi chipertext "Xnory Ynqqre". Langkah-langkah algoritma string matching yang dikombinasikan dengan algoritma ROT13 pada penelitian ini yaitu, kata pattern dienskripsi kemudian dicocokan dengan data yang ada didatabase kemudian dilakukan pencarian string menggunakan algoritma string matching dan menghasilkan data yang dicari. Pada kedua algoritma dilakukan pengujian waktu dan jumlah kecocokan antara pattern dan data untuk memperlihatkan performa kedua algoritma dengan menggunakan 200 sampai 500 data. Hasil dari pengujian 200 data untuk pencarian kata Tower membutuhkan waktu 0.00513 sec dan jumlah kecocokan 65x, 300 data untuk pencarian kata Tower membutuhkan waktu 0.00716 sec dan jumlah kecocokan 164x, 400 data untuk pencarian kata Tower membutuhkan waktu 0.01372 sec dan jumlah kecocokan 262x dan 500 data untuk pencarian kata Tower membutuhkan waktu 0.02233 sec dan jumlah kecocokan 360x. Dari hasil pengujian pada penelitian ini peneliti penyimpulkan algoritma string matching yang dikombinasikan dengan algoritma ROT13 semakin banyak data akan semakin menambah waktu walaupun masih relative kecil.

Kata kunci*:Dokumen, DMS, Algoritma ROT13, Algoritma String Matching*

Jurnal Sains Komputer & Informatika (J-SAKTI)
Volume 6 Nomor 1, Maret 2022, pp. 126-135
ISSN: 2548-9771/EISSN: 2549-7200
https://tunasbangsa.ac.id/ejurnal/index.php/jsakti

1. PENDAHULUAN

Perkembangan teknologi pada era modern ini sudah sangat pesat. Teknologi dapat dipergunakan di berbagai macam bidang salah satunya dalam bidang penataan atau penyimpanan dokumen dalam bentuk digital (*softcopy*). Penggunaan teknologi sebagai penyimpanan dokumen dapat memudahkan pencarian kembali dokumen yang dibutuhkan serta menimalisirkan kehilangan dokumen dan menjaga kerahasiaan dokumen. Penyimpanan dokumen dengan cara manual membutuhkan waktu dan tempat yang lebih untuk menyimpan *hard* dokumen hingga beresiko kehilangan dokumen akibat kelalaian penyimpanan. Dari permasalahan tersebut peneliti mencoba membuat sebuah aplikasi document management system untuk mengkontrol semua dokumen secara digital dan dapat diakses dimanapun dan kapanpun. Pada penelitian ini pengembangan aplikasi DMS untuk fitur pencarian dokumen menggunakan algoritma *string matching* dan untuk menjaga kerahasian dokumen menggunakan algoritma ROT13.

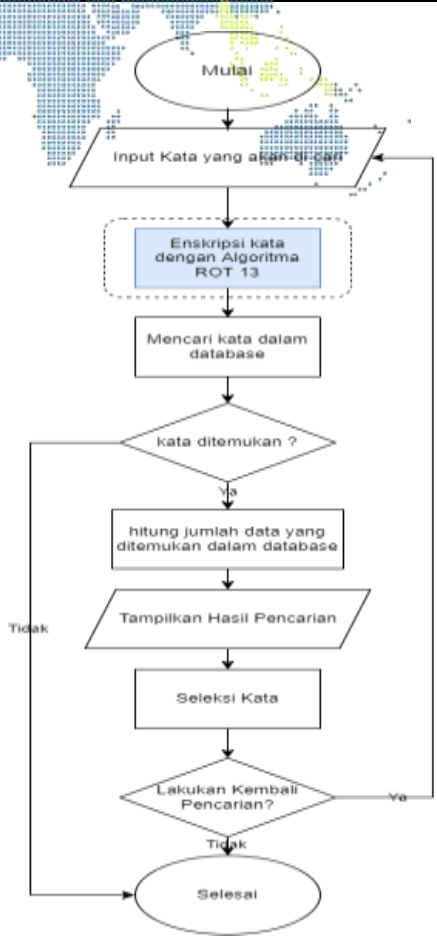
String matching merupakan proses pencarian sebuah *string* yang terdiri dari *pattern* terhadap karakter teks yang di cari[1]. Metode *string matching* yang merupakan bagian dalam proses pencarian string memegang peranan penting untuk mendapatkan dokumen yang sesuai dengan kebutuhan informasi dengan lebih cepat[2]. Belakangan ini seringkali terjadinya pencurian file yang sangat penting, dimana mereka mencari file yang mereka butuhkan yang sesuai dengan keinginan mereka. Dengan tidak adanya pengamanan sebuah judul atau nama file tersebut mereka akan sangat mudah menemukan judul file yang mereka inginkan[3]. Untuk menghindari pencurian file pada penelitian ini maka diterapkan algoritma ROT13 untuk keamanan file serta data yang terdapat didalam *database* meskipun keamanan ini tidak di desain untuk kemanan tingkat tinggi. Dan penerapan algoritma string matching guna mencari dokumen dengan lebih cepat.

Penelitian sebelum algoritma *string matching* memiliki hasil pencarian yang cepat. Hasil penelitian sebelumnya menggunakan kelipatan 100 data untuk pengujiannya dan hasil pengujian 100 data mendapatkan 50ms, 200 data mendapatkan 100ms, 300 data mendapatkan 140ms, 400 data mendapatkan 180ms dan 500 data mendapatkan 204ms[1]. Penelitian ini menggunakan *platform* website dengan menggunakan algoritma string matching sebagai pencarian dokumen, algoritma ROT13 sebagai enskripsi file dokumen dan *database* agar tidak mudah untuk mengakses dokumen secara langsung. Penelitian ini diuji mulai dari 200 data hingga 500 data untuk melihat hasil kombinasi algoritmanya.

2. METODOLOGI PENELITIAN

Pada penelitian ini, algoritma *string matching* digunakan untuk mencari sebuah nama file dokumen yang terdapat pada *database*. Pencarian kata di kombinasi dengan algoritma ROT13 untuk *mengenskripsi* kata dari *pattern* untuk dicocokan dengan data pada *database*.

Jurnal Sains Komputer & Informatika (J-SAKTI)
Volume 6 Nomor 1, Maret 2022, pp. 126-135
ISSN: 2548-9771/EISSN: 2549-7200
https://tunasbangsa.ac.id/ejurnal/index.php/jsakti



Gambar 1. alur kombinasi algoritma ROT13 dan algoritma string matching

Gambar 1 menunjukan alur kerja kombinasi algoritma ROT13 dengan algortima *string matching*. Langkah awal memasukan sebuah kata pada form yang disediakan. Kemudia kata di lakukan pengubahan menjadi *chipertext* dengan algoritma ROT13. Setelah kata diubah dilakukan pencocokan dengan *database*. Jika kata ditemukan selanjutkan akan dilakukan *matching string* menggunakan algoritma *string matching* dengan menghitung jumlah kata yang ditemukan dan menampilkan hasil selanjutnya di seleksi kata jika tidak terdapat maka tidak akan ditampilkan.

2.1. Algoritma ROT13

Pada penelitian ini algoritma ROT13 digunakan untuk *mengenskripsi* data yang ada di *database*. Penggunaan algoritma ini hanya untuk mengubah huruf *alphabet* sehingga angka dan *symbol* tidak akan berubah berbeda dengan penggunaan ROT13 pada penelitian sebelumnya[3]. Rumus matematis mengubah *plaintext* menjadi *chiperteks* sebagai berikut.

$$C_i = E(P_i) = (P_i + 13) \quad (1)$$

Pada penelitian [3] rumus matematis untuk mengubah *plaintext* menjadi *chiperteks* sebagai berikut.

$C_i = E(P_i) = (P_i + 13) mod\ 26$ (2)

Sedangkan matematis dalam mengubah *chiperteks* menjadi *chiperteks* sebagai berikut.

$P_i = D(C_i) = (C_i - 13) mod\ 26$ (3)

Implementasi algoritma ROT13 pada penelitian ini sebagai berikut:

a) Pembuatan *function* algoritma ROT13 pada file *helper codeigniter* dengan tujuan agar *function* dapat dengan mudah di panggil atau diakses disemua file.
b) Membuat sebuah *variable* string untuk menampung kata yang nantinya akan diubah menjadi *chiperteks*
c) Memecah *variable string* menjadi perkarakter dan memasukan ke dalam *array*.
d) Melakukan perulangan / *looping* untuk mengubah karakter *array* dengan mengubah dari *char* ke *decimal* dan menambahkan karakter *array* dengan + 13(P_i + 13).
e) Setelah pengubahan karakter telah selesai sampai akhir maka *variable* array di *join* untuk dijadikan sebuah kalimat *string* (*chipertext*).

Tabel 1. *Index* algoritma ROT K=13

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M

Tabel 1 menunjukan pergeseran huruf *alphabet* yang dirancang pada penelitian menggunakan algoritma ROT13 dimana K = 13 yang artinya *index* huruf di geser sejauh 13 kedepan.

2.2. Algoritma String Matching

Algoritma string matching merupakan bagian dalam proses pencarian string memegang peranan penting untuk mendapatkan dokumen yang sesuai dengan kebutuhan informasi dengan lebih cepat. Algoritma String matching adalah proses pencocokan sebuah *string* bila terjadi ketidak samaan pada saat *pattern* sejajar dengan teks (i…i + n-1), kita bisa menganggap ketidak samaan pertama terjadi di antara teks (i+j) dan *pattern* (j), dengan <j<n. berarti *text* (i…i + j) = *pattern* (0…j + 1) dan a = *text* (i+j) tidak sama dengan b = *pattern* (j), ketika kita menggeser. Implementasi string *matching* dalam penelitian ini sebagai berikut:

a) Menerima kata *pattern* dari inputan.
b) Kata *pattern* diubah menjadi *chipertext* dengan menggunakan algoritma ROT13.
c) Melakukan *query* ke *database* untuk mencari data sesuai dengan *pattern*
d) Melakukan pencocokan kata dari *database* dengan kata *pattern*.
e) Membuat *variable* untuk menampung jumlah kata yang ditemukan.

Contoh pencarian kata *pattren* sure pada teks pressure gauge

Teks : PRESSURE GAUGE

Jurnal Sains Komputer & Informatika (J-SAKTI)
Volume 6 Nomor 1, Maret 2022, pp. 126-135
ISSN: 2548-9771/EISSN: 2549-7200
https://tunasbangsa.ac.id/ejurnal/index.php/jsakti

Pattern : SURE

Index	0	1	2	3	4	5	6	7	8	9	0	10	11	12
Teks	P	R	E	S	S	U	R	E		G	A	U	G	E
Pattern	S	U	R	E										

Keterangan : *pattern* dengan *index* ke 0 tidak cocok dengan teks *index* ke 0 maka akan dilakukan pergeseran ke kanan.

Index	0	1	2	3	4	5	6	7	8	9	0	10	11	12
Teks	P	R	E	S	S	U	R	E		G	A	U	G	E
Pattern		S	U	R	E									

Keterangan: *pattern* dengan *index* ke 0 tidak cocok dengan teks *index* ke 1 maka dilakukan pergeseran kenan.

Index	0	1	2	3	4	5	6	7	8	9	0	10	11	12
Teks	P	R	E	S	S	U	R	E		G	A	U	G	E
Pattern			S	U	R	E								

Keterangan: *pattern* dengan *index* ke 0 tidak cocok dengan teks *index* ke 2 maka dilakukan pergeseran ke kanan.

Index	0	1	2	3	4	5	6	7	8	9	0	10	11	12
Teks	P	R	E	S	S	U	R	E		G	A	U	G	E
Pattern				S	U	R	E							

Keterangan: *pattern* dengan *index* ke 0 memiliki kesamaan dengan teks *index* ke 3 namun *pattern* dengan *index* ke 1 tidak cocok dengan *pattern* teks ke 4 maka dilakukan lagi pergeseran ke kanan.

Index	0	1	2	3	4	5	6	7	8	9	0	10	11	12
Teks	P	R	E	S	S	U	R	E		G	A	U	G	E
Pattern					S	U	R	E						

Keterangan: *pattern* memiliki kesamaan dengan teks maka tidak akan dilakukan pergeseran.

3. HASIL DAN PEMBAHASAN

3.1 Hasil pengujian enskripsi

Tabel 2. Hasil Pengujian Enskripsi ROT13
Sumber : Hasil Penelitian ini, 2021

PlainText	***ChiperText***
Kabel Ladder	Xnory Ynqqre
Sprinkler Head	Fcevaxyre Unrq

Jurnal Sains Komputer & Informatika (J-SAKTI)
Volume 6 Nomor 1, Maret 2022, pp. 126-135
ISSN: 2548-9771/EISSN: 2549-7200
https://tunasbangsa.ac.id/ejurnal/index.php/jsakti

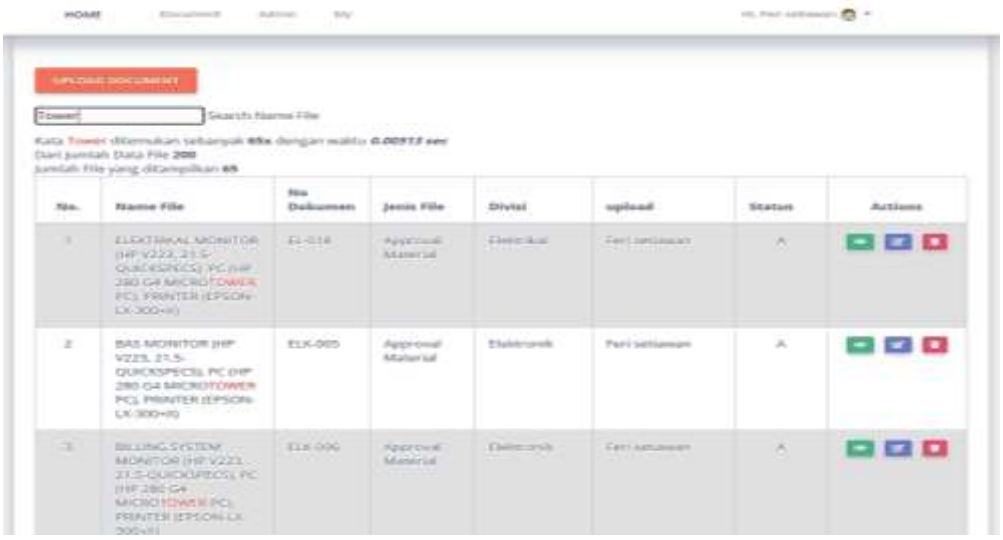
PlainText	***ChiperText***
Denah Instalasi Pipa Riser Drainage LT. B4-22 Tower Condo 2 R1	Qranu Vafgnynfv Cvcn Evfre Qenvantr YG. O4-22 Gbjre Pbaqb 2 E1
Indoor Hydrant IHB Condo 2	Vaqbbe Ulqenag VUO Pbaqb 2
Kabel Tegangan Rendah	Xnory Grtnatna Eraqnu

Tabel 2 menunjukan hasil dari ROT13 untuk *enskripsi* kata dengan menggeser index sejauh 13 ke depan. Penggeseran ini berdasarkan urutan *decimal* pada tabel ASCII. Enskripsi pada penelitian ini digunakan untuk mengenskripsi data pada *database* dan file dokumen pada *directory*.

3.2. Hasil pengujian pencarian dengan kombinasi ROT13 dan *string matching*

Untuk memperlihatkan kinerja dari algoritma *string matching* yang dikombinasikan dengan algoritma ROT13 maka pada penelitian ini dilakukan pengujian jumlah pencocokan dan waktu proses pencocokan untuk menunjukan kecepatan algoritma bekerja.

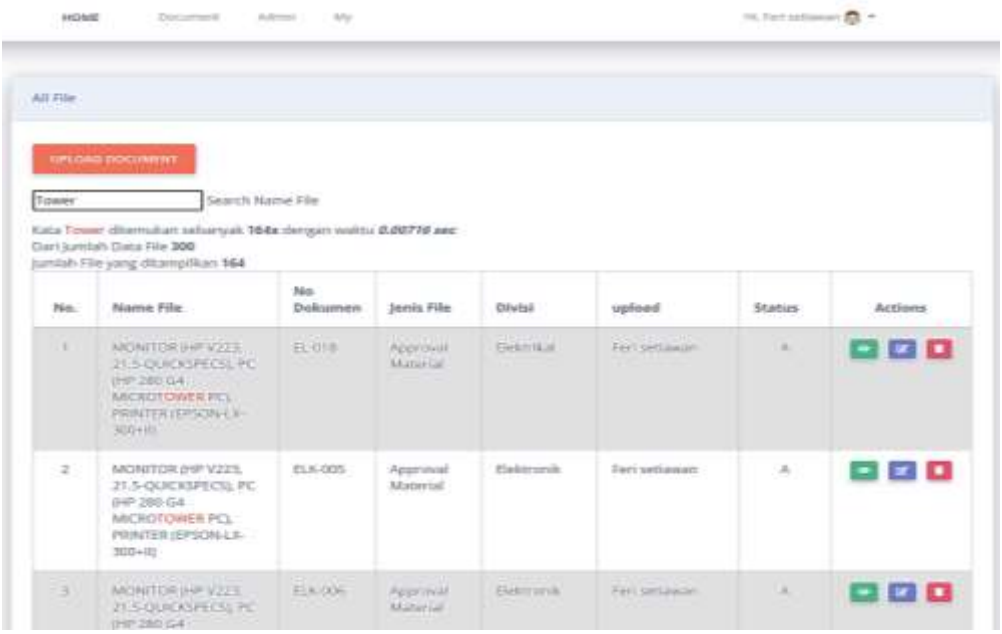
Algoritma string matching ini dikombinasi dengan algoritma ROT13 memiliki cara kerja dengan melakukan *enskripsi* data dari inputan menggunakan ROT13 kemudian mengambil data dari *database*, lalu masing-masing data dari *database* dan data dari inputan dipecah menjadi *array* dan dilakukan pengulangan untuk proses pencocokan data antara *pattern* dengan data dari *database* dari kiri kenan atau dari *index* 0 sampai ke n.



Gambar 2. Pencarian kata *Tower* dari 200 Data

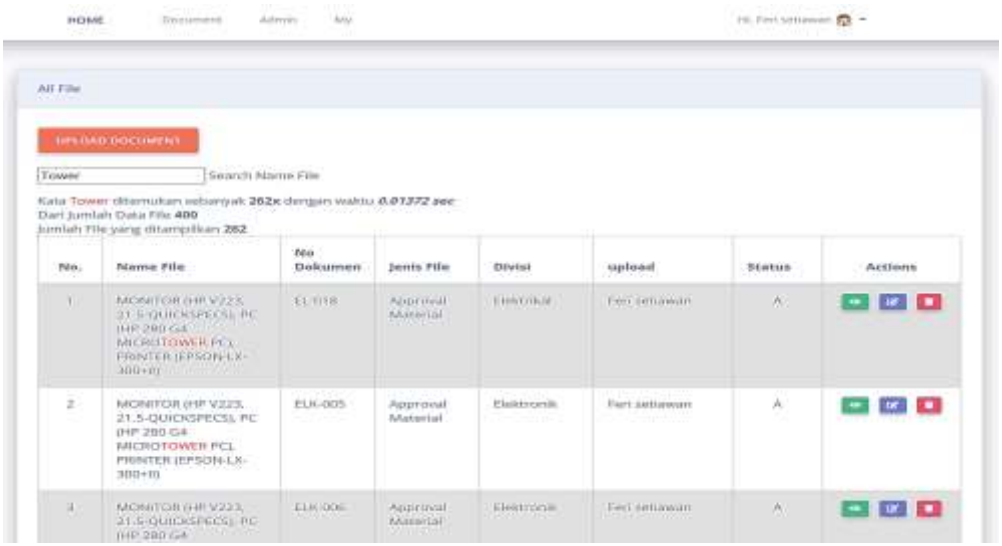
Gambar 2 menunjukan hasil pencarian kata *Tower* dari 200 data file yang terdapat pada *database*. Kecocokan kata *Tower* ditemukan sebanyak 65x dan 65 data file yang memiliki kata *Tower* ditampilkan dengan kecepatan waktu 0.00513 *sec*.

Jurnal Sains Komputer & Informatika (J-SAKTI)
Volume 6 Nomor 1, Maret 2022, pp. 126-135
ISSN: 2548-9771/EISSN: 2549-7200
https://tunasbangsa.ac.id/ejurnal/index.php/jsakti



Gambar 3. Pencarian kata *Tower* dari 300 data

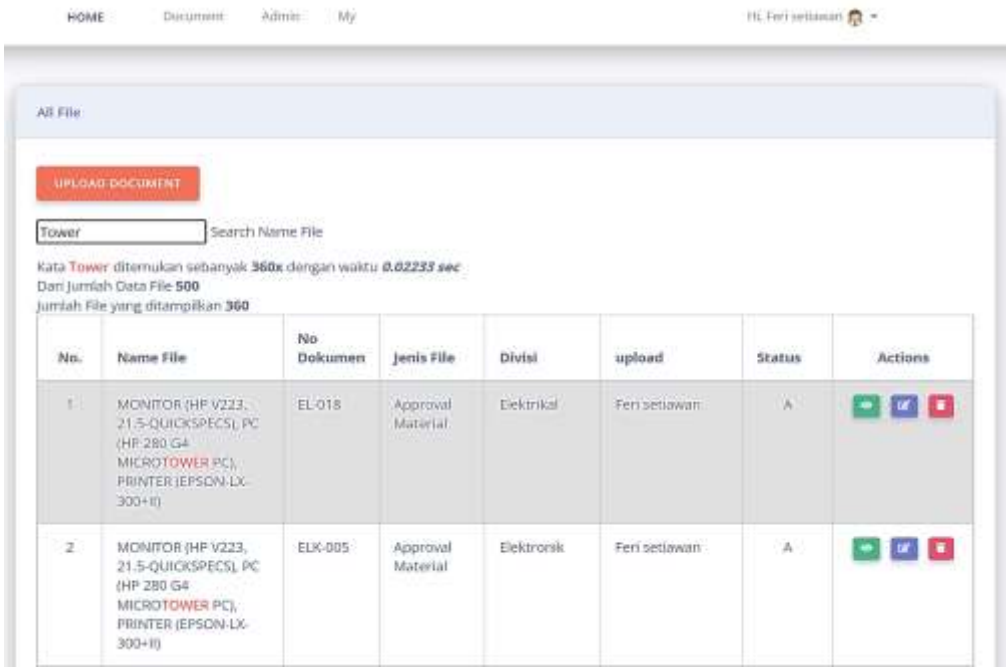
Gambar 3 menunjukan hasil pencarian kata *Tower* dari 300 data file yang terdapat pada *database*. Kecocokan kata *Tower* ditemukan sebanyak 164x dan 164 data file yang memiliki kata *Tower* ditampilkan dengan kecepatan waktu 0.00716 *sec*.



Gambar 4. Pencarian kata *Tower* dari 400 data

Gambar 4 menunjukan hasil pencarian kata *Tower* dari 400 data file yang terdapat pada *database*. Kecocokan kata *Tower* ditemukan sebanyak 262x dan 262 data file yang memiliki kata *Tower* ditampilkan dengan kecepatan waktu 0.01372 *sec*.

Jurnal Sains Komputer & Informatika (J-SAKTI)
Volume 6 Nomor 1, Maret 2022, pp. 126-135
ISSN: 2548-9771/EISSN: 2549-7200
https://tunasbangsa.ac.id/ejurnal/index.php/jsakti



Gambar 5. Pencarian kata *Tower* dari 500 data

Gambar 5 menunjukan hasil pencarian kata *Tower* dari 500 data file yang terdapat pada *database*. Kecocokan kata *Tower* ditemukan sebanyak 360x dan 360 data file yang memiliki kata *Tower* ditampilkan dengan kecepatan waktu 0.02233 *sec*.

Tabel 3. Hasil Pencarian kata *Tower* yang terdapat pada *database*
(sumber : Hasil penguji peneliti, 2021)

Jumlah Data	kata	Ditemukan	Jumlah File	Waktu
200	*Tower*	65x	65	0.00513 sec
300	*Tower*	164x	164	0.00716 sec
400	*Tower*	262x	262	0.01372 sec
500	*Tower*	360x	360	0.02233 sec

Tabel 3 menunjukan hasil pengujian pencarian kata menggunakan Algoritma string matching dikombinasikan dengan algoritma ROT13 membutuhkan waktu yang cukup singkat dalam pemprosesan *string matching*. hal ini sesuai dengan penelitian sebelumnya [1]. Pada algoritma string matching semakin banyak data yang digunakan maka akan menambahkan waktu yang dibutuhkan tetapi masih relatif kecil.

Tabel 4. Hasil Pengujian Pencarian Kata yang terdapat pada database
(sumber : Hasil penguji peneliti, 2021)

Jumlah Data	Kata	Ditemukan	Jumlah File	Waktu
500	Instalasi	310x	310	0. 02148 *sec*
500	*Shop*	280x	280	0. 01729 *sec*

Jumlah Data	Kata	Ditemukan	Jumlah File	Waktu
500	*Tower*	360x	360	0. 02233 *sec*
500	Area	114x	113	0. 00885 *sec*
500	Pipa	179x	169	0.01032 *sec*
500	Denah	148x	148	0.00902 *sec*

Tabel 4 menunjukan hasil pengujian pencarian kata dengan menggunakan kedua kombinasi algoritma menghasilkan waktu yang relative singkat. Penelitian ini algoritma string matching sangat *sensitive* dalam pencarian kata. Jika kata *pattern* memiliki salah satu huruf kapital dan di data tidak ada huruf kapital maka sistem algoritma ini tidak mendeteksi ada kecocokan kata dengan kata lain akurasinya tidak 100%. Pada penelitian ini agar akurasi pencarian kata 100% peneliti mengubah kata *pattern* dan kata *text* menjadi huruf kecil agar mendapatkan hasil akurasi yang 100%.

4. SIMPULAN

Penelitian ini menghasilkan sistem pencarian dokumen dengan algoritma *string matching* yang dikombinasi dengan algoritma ROT13 dan sistem enskripsi menggunakan algoritma ROT13. Dengan adanya kedua algoritma tersebut ini dapat memudahkan pencarian file dokumen berdasarkan *string* secara akurat, serta memberikan keamanan dokumen walaupun tingkat keamanannya tidak didesain dengan tingkat tinggi. penggunaan algoritma *string matching* pada penelitian ini memiliki kekurangan dalam penggunaan huruf *lowercase* dan *uppercase* sehingga sebelum melakukan pencarian kedua data diubah menjadi *lowercase* atau *uppercase* sehingga algoritma *string matching* ini menghasilkan tingkat akurasi yang maksimal. Hasil penelitian ini dalam pencarian kata *Tower* dari 200 data pada *database* mendapatkan 56x kecocokan kata dengan waktu 0.00513 *sec*. Pencarian kata *Tower* dari 300 data pada *database* mendapatkan 164x kecocokan kata dengan waktu 0.00716 *sec*. pencarian kata *Tower* dari 400 data pada *database* mendapatkan 262x kecocokan kata dengan waktu 0.01372 *sec* dan pencarian kata *Tower* dari 500 data pada *database* mendapatkan 360x kecocokan dengan waktu 0.02233 *sec*.

DAFTAR PUSTAKA

[1] I. Ahmad, R. Indra Borman, G. G. Caksana, and J. Fakhrurozi, "Implementasi String Matching Dengan Algoritma Boyer-Moore Untuk Menentukan Tingkat Kemiripan Pada Pengajuan Judul Skripsi/TA Mahasiswa (STUDI KASUS: UNIVERSITAS XYZ)," [Online]. Available: https://doi.org/10.31598.

[2] I. Mulyawati, R. Subagio, and D. Martha, "Implementasi Metode String Matching Untuk Aplikasi Pengarsipan Dokumen (Studi Kasus: Smpn 3 Sumber Kab. Cirebon)," *J. Digit*, vol. 7, no. 1, pp. 50–61, 2017.

[3] Hendrik, "Kombinasi Algoritma Huffman dan Algoritma ROT 13 Dalam Pengamanan File Docx," *J. Inf. Syst. Res.*, vol. 2, no. 1, pp. 40–46, 2020.

[4] A. A. Nababan and M. Jannah, "Algoritma String Matching Brute Force Dan Knuth-Morris-Pratt Sebagai Search Engine Berbasis Web Pada Kamus Istilah Jaringan Komputer," *J. Mantik Penusa*, vol. 3, no. 2, pp. 87–94, 2019.

[5] A. S. Sumi, P. Purnawansyah, and L. Syafie, "Analisa Penerapan Algoritma Brute Force Dalam Pencocokan String," *Pros. SAKTI (Seminar Ilmu Komput. dan Teknol. Informasi)*, vol. 3, no. 2, pp. 88–92, 2018, [Online]. Available: http://e-journals.unmul.ac.id/index.php/SAKTI/article/view/1836.

[6] G. F. H. Nainggolan, S. Andryana, and A. Gunaryati, "Pencarian Berita Pada Web Portal Menggunakan Algoritma Brute Force String Matching," *JIPI (Jurnal Ilm. Penelit. dan Pembelajaran Inform.*, vol. 6, no. 1, pp. 1–10, 2021, doi: 10.29100/jipi.v6i1.1824.

[7] R. M. Aresta, E. W. Pratomo, V. Geraldino, J. D. Santoso, and S. Mulyatun, "Implementasi Multi Enkripsi Rot 13 Pada Symbol Whatsapp," *J. Inf. Syst. Manag.*, vol. 2, no. 1, pp. 1–5, 2020,

[8] H. N. Buluş, E. Uzun, and A. Doruk, "Comparison of string matching algorithms in web documents," *Int. Sci. Conf.*, vol. 2, no. November, pp. 279–282, 2017, [Online]. Available: https://erdincuzun.com/wp-content/uploads/download/s5_p256.pdf.

[9] M. Bhagya Sri, R. Bhavsar, and P. Narooka, "String Matching Algorithms," *Int. J. Eng. Comput. Sci.*, vol. 7, no. 03, pp. 23769–23772, 2018, doi: 10.18535/ijecs/v7i3.19.

[10] A. Afandi *et al.*, "Perbandingan Algoritma Boyer Moore dan Algoritma Brute Force pada Aplikasi Hadits Berbasis Android," vol. 2, no. 1, pp. 1–2, 2017, doi: 10.15575/join.