Modifikasi Metode *Base64* Menggunakan *Caesar Cipher* Dan Kunci Rahasia

Ripto Sudiyarno*
[1]Magister Teknik Informatika, Universitas Amikom Yogyakarta; Jalan Ring Road Utara, Yogyakarta, Telp. (0274) 884201
e-mail: ***ripto.s@students.amikom.com**

Abstrak

Seiring perkembangan teknologi yang sangat pesat, keamanan pesan merupakan hal yang sangat penting sekali dalam menjaga sebuah kerahasiaan terutama pada pesan yang dianggap sensitif agar tidak mudah dibaca oleh pihak yang memiliki kewenangan, agar mencegah adanya kebocoran pesan sensitive ini dilakukan langkah enkripsi dengan kunci rahasia sehingga pesan tidak dapat dibaca sebelum dibuka kembali menggunakan kunci sebelumnya yang diberikan. Pada penelitian ini melakukan enkripsi pesan menggunakan modifikasi terhadap algoritma Base64. Algoritma base64 merupakan algoritma dengan format ASCII untuk encoding dan decoding yang didasarkan pada 64 bit bilangan dasar. Hasil dari penelitian ini dapat membuktikan bahwa modifikasi terhadap algoritma base64 dapat menyandikan pesan dengan cepat serta dapat merubah kembali kebentuk semula tanpa merubah keaslian pesan yang input sebelumnya.

Kata kunci*— Algoritma base64, Caesar Cipher, Kriptografi, Modifikasi.*

1. PENDAHULUAN

Seiring perkembangan teknologi yang sangat pesat, keamanan pesan merupakan hal yang sangat penting sekali dalam menjaga sebuah kerahasiaan terutama pada pesan yang dianggap sensitif agar tidak mudah dibaca oleh pihak yang memiliki kewenangan. Lebih lanjut, untuk mencegah adanya kebocoran pesan sensitif, perlu dilakukan langkah enkripsi dengan kunci rahasia sehingga pesan tidak dapat dibaca sebelum dibuka kembali menggunakan kunci sebelumnya yang diberikan. Penelitian yang telah dilakukan oleh Mochammad Firman Arif menunjukkan bahwa algoritma *base64* dan *rotation13* yang diimplementasikan pada URL membuat website lebih aman karena celah keamanan dapat ditutup, setelah melalui proses penggabungan kedua algoritma tersebut. Namun, penggabungan kedua algoritma tersebut dianggap memiliki keamanan yang kurang tinggi [1]. Penelitian lainnya yang melakukan pengujian sistem pada sistem keamanan basis data klien di P.T. Infokes dengan kriptografi kombinasi *RC4* dan *Base64* menunjukkan bahwa sistem yang diusulkan dapat mengamankan *database* serta pengujian *white* dan *black box testing* menunjukan proses *upload* menggunakan metode yang diusulkan pada *database* berjalan dengan baik, tetapi tidak adanya kunci rahasia yang diusulkan akan sangat besar kemungkinan untuk proses masih bisa disadap [2].

Algoritma *Base46* telah digunakan untuk mengekripsi *file text* kedalam bentuk acak dan dapat dikembalikan ke bentuk semula. Namun, algoritma tersebut masih memiliki kekurangan jika menggunakan *file text* yang kurang atau sama dengan 6 karakter [3]. Penerapan lainnya yaitu algoritma ROT13 dan ElGamal untuk mengenkripsi dan dekripsi pesan rahasia terhadap 10 mahasiswa melalui pengambilan angket didapatkan nilai sebesar 83,75% yang mengindikasikan bahwa kedua algoritma ini sangat layak digunakan untuk mengamankan pesan rahasia [4]. Salah satu hal yang penting dalam sebuah pesan yang sensitif adalah adanya kerahasiaan, yang memiliki nilai yang bervariasi setiap orang. Oleh karena itu, penelitian ini mengusulkan peningkatan pada algoritma *base64* dengan kunci rahasia yang support dalam bentuk apapun (*unicode*). Perancangan yang dilakukan pada penelitian ini, diharapkan dapat

membantu mengamankan pesan dan sulit untuk dibaca oleh pihak yang tidak berwenang jika tidak memiliki kunci serta proses.

2. METODE PENELITIAN

Teknik pengumpulan data pada perancangan peningkatan pada algoritma *base64* ini menggunakan penelitian kepustakaan (*Library Search*) yaitu metode yang digunakan dengan mencari data yang bersangkutan dan memiliki kesamaan atau korelasi dengan penelitian yang sedang dilakukan dan juga untuk penambahan referensi.

2.1 Keamanan Data

Pada masa ini hampir semua bidang membutuhkan bantuan komputer, dan juga tentu semuanya mempunyai data penting sesuai bidang yang bersangkutan. Oleh karena itu, memperhatikan sisi keamanan data atau informasi itu sangat penting terlepas dari banyaknya cara bagi pihak yang tidak berkepentingan untuk mengakses data tersebut. Pihak yang berwenang juga punya beberapa cara untuk mengamankannya. Keamanan data pada jaringan merupakan hal yang sangat dibutuhkan untuk menjaga privasi, agar data yang dikirimkan tidak sampai ke tangan orang yang tidak bertanggung jawab serta disembunyikan menggunakan algoritma kriptografi [5]–[7].

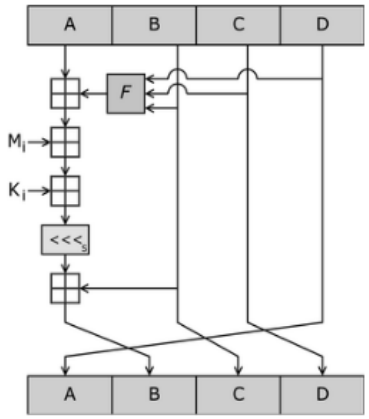
2.2 Algoritma Base64

Character yang dikeluarkan pada proses *encoding* algoritma ini terdiri dari a-z, A-Z, 0-9, +, / dan yang terakhir yaitu (=) sebagai sebuah penyeimbang atau disebut juga *pad*. Algoritma ini didasari pada bilangan dasar *binary*. Teknik *encoding* algoritma *base64* ini sederhana seperti berikut [8], [9].

a. Pecah string *bytes* tersebut menjadi per-3 *bytes*.
b. Selanjutnya satukan 3 *bytes* menjadi 24 *bit*.
c. Langkah selanjutnya 24 *bit* yang tersimpan di *buffer* dibagi ke dalam 6 *bit*, maka akan menghasilkan 4 bagian.
d. Masing-masing bagian diubah kedalam bentuk *decimal*.
e. Nilai desimal tersebut dijadikan indeks untuk memilih karakter penyusun terhadap *base64* dengan nilai akhir indeks ke 64.

2.3 Algoritma MD5

Algoritma MD5 merupakan algoritma *message-digest* yang diperkenalkan oleh seorang Profesor Ronal Rivest pada tahun 1991. Algoritma ini memiliki panjang 32 digit dalam bentuk *heksa decimal* [10], [11]. Algoritma ini memproses teks input ke dalam bentuk *binary* digit sebanyak 512 bit, kemudian dibagi ke dalam 32 bit sub blok sebanyak 16 blok. Keluaran dari algoritma MD5 berupa 4 buah blok yang masing-masing 32 bit, dimana semuanya berjumlah 128 bit dan disebut sebagai nilai fungsi *hash*. Satu operasi algoritma MD5 terdiri dari 64 operasi, dan dikelompokkan dalam empat putaran dari 16 operasi. F adalah sebuah fungsi nonlinear. Satu fungsi digunakan pada tiap-tiap putaran, menunjukkan blok 32 bit dari masukan pesan dan menunjukkan konstanta 32 bit yang berbeda untuk tiap-tiap operasi. Berikut ini adalah model atau desain dari algoritma MD5 [12].



Gambar 1 Model design algoritma MD5

2. 4 Algoritma Caesar Cipher

Algoritma c*aesar cipher* merupakan algoritma yang digunakan pada masa kaisar romawi yaitu Julius Caesar. Algoritma ini bekerja dengan cara mensubtitusikan setiap karakter huruf dengan huruf lainnya dengan posisi pergeseran yang sudah ditentukan [9], [13].

Secara matematis dapat dituliskan sebagai berikut:

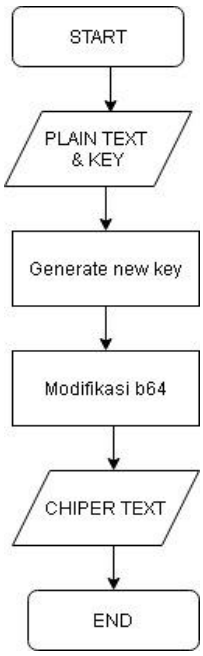
$$c_i = E\,(p_i) = \,(p_i + k)\ mod\ 26 \quad (1)$$

$$p_i = D\,(c_i) = \,(c_i - k)\ mod\ 26 \quad (2)$$

Persamaan (1) adalah rumus *encoding* dan persamaan (2) adalah sebaliknya yaitu *decoding*. Dikarenakan abjad berjumlah 26 huruf, maka setiap pergeseran dapat dilakukan dari index 0-25. Algoritma Caesar Cipher tergolong kepada algoritma subtitusi monoalfabetik, karena setiap huruf yang sama akan digantikan dengan huruf yang sama disepanjang index yang ada.

3. HASIL DAN PEMBAHASAN

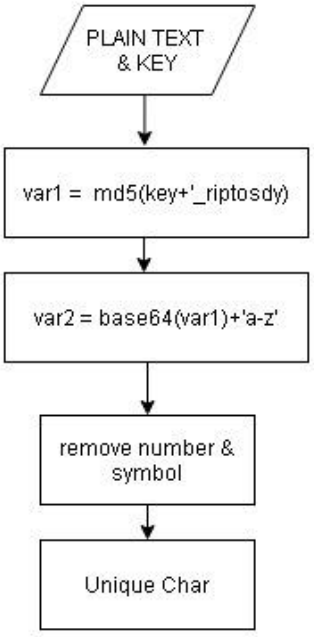
Pada penelitian ini dilakukan enkripsi dan dekripsi pesan menggunakan modifikasi algoritma *base64* [13] seperti terlihat pada Gambar 2.



Gambar 2 Proses Enkripsi menggunakan modifikasi algoritma *base64*

3.1 Tahap Enkripsi

Pada proses pertama dilakukan inputan *plain text* dan juga kunci, selanjutnya kunci yang diinputkan akan digunakan untuk membangkitkan identitas baru untuk *key* baru yang akan digunakan pada proses enkripsi dengan proses ditampilkan oleh Gambar 3.



Gambar 3 Proses pembangkitan kunci baru

Tahapan dari proses pada Gambar 3 dijelaskan sebagai berikut:

1. *Key* yang diinputkan akan ditambah dengan *static key* yaitu *'_riptosdy'* selanjutnya dienkripsi menggunakan algoritma MD5.
2. Hasil dari proses pertama akan di encoding menggunakan algoritma *base64* serta ditambah dengan '*abcdefghijklmnopqrstuvwxyz*'.
3. Proses selanjutnya dilakukan penghapusan terhadap angka dan simbol pada proses sebelumnya.
4. Langkah terakhir dengan cara split tiap karakter pada string, kemudian dilanjutkan dengan proses eliminasi terhadap karakter yang berulang sehingga kunci baru berbentuk karakter *unique*.

Pada proses selanjutnya dilakukan enkripsi dengan kunci pada proses sebelumnya seperti berikut.

1. Menghitung panjang kunci.
2. *Plain text* yang diinputkan akan di encoding menggunakan algoritma *base64*.
3. Dilanjutkan dengan proses pergeseran maju terhadap string menggunakan metode *caesar cipher* sebanyak panjang kunci yang sebelumnya didapat.
4. kemudian di encoding lagi menggunakan algoritma *base64*.
5. Selanjutnya akan didapati hasil dengan mengganti semua karakter A-Z dengan karakter yang ada pada kunci baru.

3.2 Tahap Dekripsi

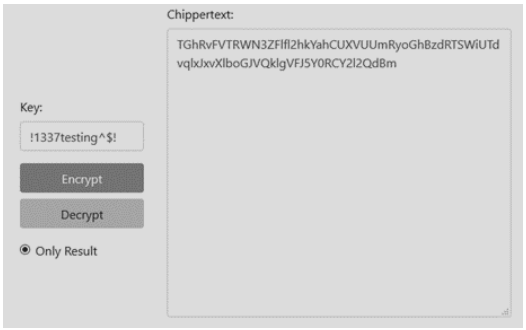
Proses membangkitkan kunci baru pada prosedur Dekripsi masih sama dengan langkah yang sudah dijelaskan pada Gambar 3. Selanjutnya dilakukan proses Dekripsi sebagai berikut:

1. *Cipher text* yang diinput akan digantikan dengan kunci yang baru dengan a-z.

2. Selanjutnya proses pergeseran mundur terhadap string menggunakan metode *caesar cipher* sebanyak panjang kunci yang sebelumnya didapat.
3. Kemudian di encoding lagi menggunakan algoritma *base64*.
4. Hasil dari proses sebelumnya akan di decoding menggunakan algoritma *base64*.
5. *Plain text*.

3.3 Implementasi pada Aplikasi berbasis WEB

Implementasi dari algoritma *caesar cipher* pada aplikasi web dapat dilihat seperti pada Gambar 4.



Gambar 4 *Interface* aplikasi berbasis web

Pada Gambar 4 diinputkan pesan dan key sebelum proses enkripsi sebagai berikut:

Pesan : Beneran ini pesan rahasia tolong dijaga hehe!

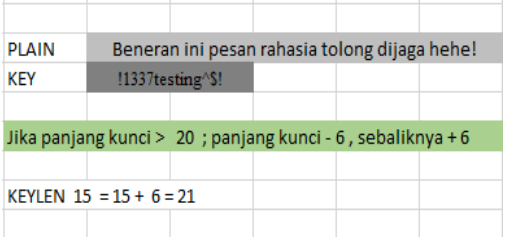
Key : !1337testing^$!

Setelah dilakukan proses enkripsi didapatkan *cipher text* "TGhRvFVTRWN3ZFlfl2hkYahCUXVUUmRyoGhBzdRTSWiUTdvqlxJxvXlboGJVQklgVFJ 5Y0RCY2l2QdBm". Lebih lanjut, proses dekripsi juga berjalan tanpa *error* dan berhasil mengembalikan text pesan yang ditampilkan oleh Gambar 5.

Gambar 5 Proses Dekripsi dan dihasilkan *Plain text*

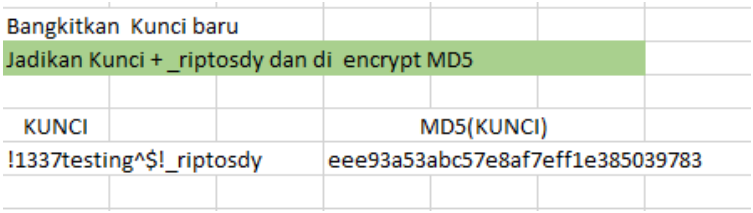
3.4 Pembahasan

Dalam proses enkripsi ini langkah-langkah yang dilakukan adalah melakukan perhitungan panjang kunci pada Gambar 6.



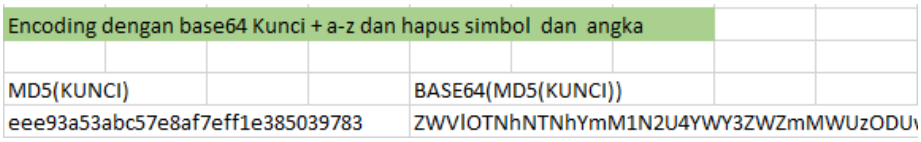
Gambar 6 Perhitungan panjang kunci

Perhitungan nilai kunci baru dihitung berdasarkan panjang string kunci yang diinputkan dan dikembalikan sesuai ketentuan seperti pada Gambar 6. Selanjutnya, pembangkitan nilai kunci baru dengan penambahan string *"_riptosdy"* pada kunci sebelumnya dan dienkripsi menggunakan algoritma MD5 seperti pada Gambar 7.



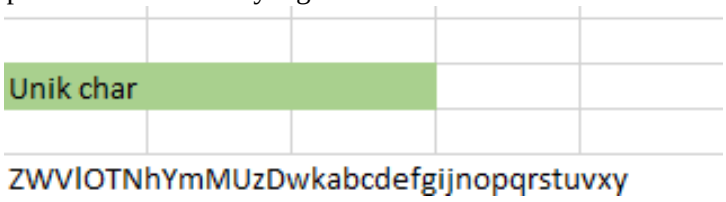
Gambar 7 Pembangkitkan kunci baru

Kunci baru yang sudah dienkripsi, selanjutnya akan diencoding menggunakan algoritma *base64* dan dilakukan penghilangan terhadap simbol dan angka.



Gambar 8 Encoding kunci baru

Gambar 9 menampilkan hasil karakter yang unik.

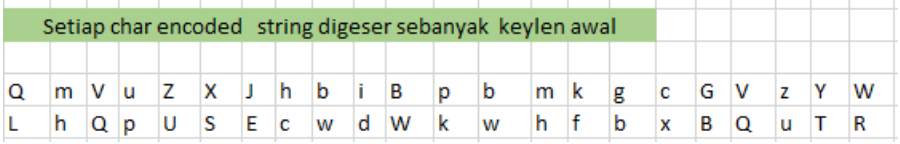


Gambar 9 Unik *character*

Tahap selanjutnya adalah mendapatkan nilai kunci baru dengan cara menseleksi unik pada setiap *character*. Setelah itu, pesan di encode menggunakan *base64* seperti pada Gambar 10.

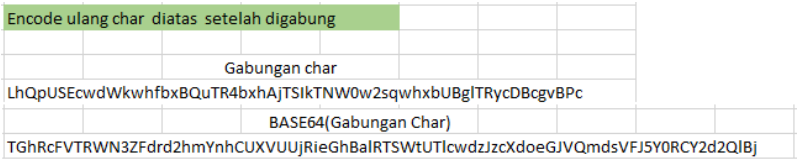
Gambar 10 Encode pesan menggunakan algoritma *base64*

Setiap *char*-nya kemudian juga akan digeser sebanyak panjang kunci yang didapat seperti yang ditampilkan oleh Gambar 11 sebanyak 21 langkah.



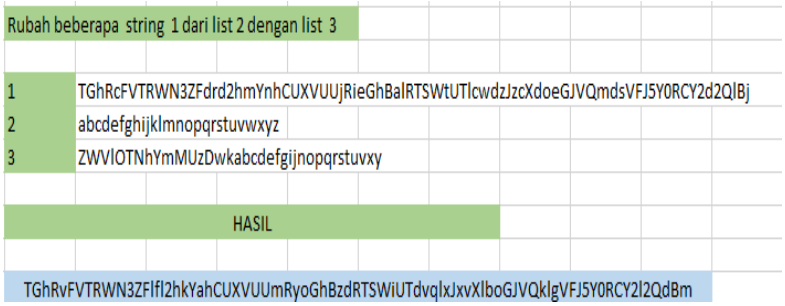
Gambar 11 Penggeseran setiap *char*

Proses berikutnya adalah menggabungkan setiap *char* dan diencoding lagi menggunakan algoritma *base64*.



Gambar 12 Penggabungan *char*

Hasil dari penggabungan tersebut, didapatkan *string* yang terlihat di Gambar 11. *String* tersebut kemudian diganti dengan *string key* baru yang sudah ada dengan berpatokan pada list nomor 2 seperti pada Gambar 13.



Gambar 13 Hasil *chipper text*

Berdasarkan prosedur encoding dan decoding algoritma *base64*, perubahan yang dilakukan pada modifikasi di dalam penelitian ini dengan cara menambahkan kunci serta menggeser setiap karakter didalam *string* sesuai dengan panjang kunci (*k*) sesuai dengan persamaan (1). Tahap selanjutnya juga melakukan penambahan dan pengurangan nilai ASCII untuk tiap-tiap karakter dengan *keychar*-nya lalu dikembalikan lagi kedalam bentuk karakter dengan nilai total sebelumnya menjadi indeks decimal terhadap *char* seperti pada Gambar 14.

```
$keychar = substr($keyLen, ($i % strlen($keyLen))-1, 1);
$char = chr(ord($char)+ord($keychar));
```

Gambar 14 Prosedur menggunakan bahasa PHP

Proses perhitungan panjang kunci dilakukan menggunakan bahasa pemrograman PHP seperti Gambar 15.

```
function getlenghtkey($key)
{
    if(strlen($key > 20))
    {
        $anu = strlen($key) - 6;
    }else
    {
        $anu = strlen($key) + 6;
    }

    return $anu;
}
```

Gambar 15 Proses Perhitungan panjang kunci dalam bahasa PHP

4. KESIMPULAN

Berdasarkan penelitian, implementasi dan pembahasan yang telah dilakukan maka kesimpulan dalam penelitian ini adalah:

1. Modifikasi algoritma *base64* yang dilakukan pada penelitian ini berjalan dengan baik.
2. Penggunaan kunci untuk *base64* akan memperulit proses dekripsinya dibandingkan dengan proses standar pada algoritma *base64* itu sendiri.

5. SARAN

Saran untuk peelitian selanjutnya dapat menambahkan beberapa metode atau algoritma lainnya agar *chipper text* dari pesan yang dienkripsi tidak mudah dibaca dan lebih aman dari penelitian ini. Penelitian selanjutnya diharapkan dapat menambahkan steganografi agar pesan yang disampaikan lebih aman.

UCAPAN TERIMA KASIH

Penulis mengucapkan terima kasih kepada Ibu Liarni yang telah memberi “dukungan finansial” terhadap penelitian ini.

DAFTAR PUSTAKA

[1] M. F. Arif, “Implementasi Enkripsi URL Pada Website Menggunakan Metode Base64 Dan Rotation13,” *J. SPIRIT*, vol. 12, no. 1, 2020.

[2] I. Afrianto and N. Taliasih, “Sistem Keamanan Basis Data Klien PT Infokes Menggunakan Kriptografi Kombinasi RC4 Dan Base64,” *J. Nas. Teknol. dan Sist. Inf.*, vol. 6, no. 1, pp. 9–18, 2020.

[3] A. Azlin, F. Musadat, and J. Nur, “Aplikasi Kriptografi Keamanan Data Menggunakan Algoritma Base64,” *J. Inform.*, vol. 7, no. 2, 2018.

[4] M. Z. Zuhri, “Implementasi Algoritma Rot13 Dan Elgamal Untuk Enkripsi Dan Dekripsi Pesan Rahasia,” *Univ. Nusant. Pgri Kediri*, 2018.

[5] F. Alfiah, R. Sudarji, and D. T. Al Fatah, “Aplikasi Kriptografi Dengan Menggunakan Algoritma Elgamal Berbasis Java Desktop Pada Pt. Wahana Indo Trada Nissan Jatake,” *ADI Bisnis Digit. Interdisiplin J.*, vol. 1, no. 1, pp. 22–34, 2020.

[6] M. Mesran and S. D. Nasution, “Peningkatan Keamanan Kriptografi Caesar Cipher dengan Menerapkan Algoritma Kompresi Stout Codes,” *J. RESTI (Rekayasa Sist. Dan Teknol. Informasi)*, vol. 4, no. 6, pp. 1209–1215, 2020.

[7] A. S. Manullang, R. Puspasari, and W. Verina, “Penyandian Database Menggunakan Metode Base64 Dan Rot13,” *J. Mhs. Fak. Tek. dan Ilmu Komput.*, vol. 1, no. 1, pp. 283–292, 2020.

[8] R. Aulia, A. Zakir, and D. A. Purwanto, “Penerapan Kombinasi Algoritma Base64 Dan Rot47 Untuk Enkripsi Database Pasien Rumah Sakit Jiwa Prof. Dr. Muhammad Ildrem,” *InfoTekJar J. Nas. Inform. dan Teknol. Jar.*, vol. 2, no. 2, pp. 146–151, 2018.

[9] D. Ariyus, *Pengantar ilmu kriptografi: teori analisis & implementasi*. Penerbit Andi, 2008.

[10] N. Hayati, M. A. Budiman, and A. Sharif, “Implementasi Algoritma RC4A dan MD5 untuk menjamin Confidentiality dan integrity pada file teks,” *SinkrOn*, vol. 1, no. 2, 2017.

[11] W. I. A. Ray, “Implementasi Metode MD5 Untuk Autentikasi Hasil Scan Citra Ijazah,” *Pelita Inform. Inf. dan Inform.*, vol. 9, no. 3, pp. 149–154, 2021.

[12] S. U. Lubis, “Implementasi Metode MD5 Untuk Mendeteksi Orisinalitas File Audio,” *KOMIK (Konferensi Nas. Teknol. Inf. dan Komputer)*, vol. 3, no. 1, 2019.

[13] D. Nurani, “Perancangan Aplikasi Email Menggunakan Algoritma Caesar CIPHER dan Base64,” *JISKA (Jurnal Inform. Sunan Kalijaga)*, vol. 2, no. 3, pp. 175–180, 2018.