Perlindungan *Web* pada *Login* Sistem Menggunakan Algoritma Rijndael

Eka Adhitya Dharmawan, Erni Yudaningtyas, dan M. Sarosa

***Abstrak*–- Pengguna internet, biasanya menggunakan fasilitas internet untuk melakukan proses pengubahan informasi. Sehingga keamanan data sangatlah penting. Kebutuhan akan informasi menjadikan para pengembang *website* menyajikan berbagai macam layanan bagi para pengguna. Namun kebanyakan dari para pengembang *website* mengabaikan keamanan sistem pada *website* tersebut. Serangan yang paling banyak digunakan oleh para penyerang tersebut adalah serangan SQL *Injection*. Penelitian ini difokuskan pada pengamanan sistem menggunakan algoritma Rijndael untuk mengenkripsi data. Algoritma Rijndael terpilih sebagai algoritma kriptografi yang dapat melindungi informasi dengan baik serta efisien dalam implementasinya dan dinobatkan sebagai *Advanced Encryption Standard* (AES). Algoritma ini akan ditanamkan pada *login* sistem untuk melindungi akses yang tidak sah dari penyerang. Hasil dari penggunaan algoritma Rinjdael dapat melindungi sistem *login* dengan baik sehingga sistem dinyatakan aman dari para penyerang *website*.**

Kata Kunci*—Rijndael, *Website, Database, SQL Injection.

I. PENDAHULUAN

MASALAH keamanan merupakan salah satu aspek penting dari sebuah sistem informasi akan tetapi masalah keamanan sering kali kurang mendapat perhatian dari para pemilik dan pengelola sistem informasi. Seringkali masalah keamanan berada di urutan kedua, atau bahkan di urutan terakhir dalam daftar hal-hal yang dianggap penting [1].

Beberapa hal penting yang perlu diperhatikan pada keamanan *web* dan menjadi masalah yang penuh kerentanan adalah *login* dan *database*. Sistem *login* yang menggunakan *database* sebagai autentikasi *user* dan *password* sangat rentan untuk diretas. SQL *Injection* adalah salah satu teknik serangan yang dapat digunakan oleh penyerang untuk mengeksploitasi aplikasi *web*, sebagai akibatnya penyerang bisa mendapatkan akses tidak sah ke *database* atau untuk mengambil informasi langsung dari *database* [2].

Keamanan data sangat dibutuhkan untuk menjaga dan melindungi kerahasiaan data keuangan. Sudah banyak kejadian tentang pembobolan sistem berbasis *web*, karena kurangnya perhatian akan keamanan sistem yang dibangun. Keamanan harus benar-benar diperhatikan oleh para pengembang *web* agar mempunyai sistem keamanan yang lebih baik dan tidak gampang diretas oleh orang-orang yang tidak berkepentingan.

Salah satu teknik pengamanan sistem adalah dengan menggunakan teknik enkripsi dan deskripsi data. Enkripsi dan dekripsi termasuk dalam bidang ilmu kriptografi. Kriptografi adalah suatu ilmu yang mempelajari bagaimana cara menjaga agar data atau pesan tetap aman saat dikirimkan, dari pengirim ke penerima tanpa mengalami gangguan dari pihak ketiga [3]. Pengamanan ini dilakukan dengan mengenkripsi informasi tersebut dengan suatu kunci khusus. Informasi ini sebelum dienkrip dinamakan *plaintext*. Setelah dienkrip dengan suatu kunci dinamakan *ciphertext* [4].

Salah satu algoritma yang ditentukan oleh *National Institute of Standards and Technology* (NIST) sebagai pemenang dalam perlombaan memperebutkan kandidat AES, terpilihlah algoritma Rijndael sebagai pemenang. Algoritma Rijndael terpilih sebagai algoritma kriptografi yang dapat melindungi informasi dengan baik serta efisien dalam implementasinya dan dinobatkan sebagai AES [4].

Rijndael termasuk dalam jenis algoritma kriptografi yang sifatnya simetri dan blok sandi. Dengan demikian algoritma ini mempergunakan kunci yang sama saat enkripsi (bentuk acak) dan dekripsi (mengembalikan ke bentuk semula) serta masukan dan keluarannya berupa blok dengan jumlah bit tertentu. Rijndael mendukung berbagai variasi ukuran blok dan kunci yang akan digunakan. Namun Rijndael mempunyai ukuran blok dan kunci yang tetap sebesar 128, 192, 256 bit. Pemilihan ukuran blok data dan kunci akan menentukan jumlah proses yang harus dilalui untuk proses enkripsi dan dekripsi.

Berbagai penelitian telah dilakukan terhadap algoritma Rijndael dan serangan SQL *Injection*. (Soyjaudah et al., 2004) melakukan penelitian tentang enkripsi yang digunakan dalam sistem komunikasi untuk melindungi informasi yang dikirim lewat saluran komunikasi agar tidak ditangkap dan dibaca oleh pihak yang tidak berwenang [5]. (Majumder dan Saha, 2009) melakukan penelitian tentang *Analysis SQL Injection Attack* [6].

Pada penelitian ini penulis akan membuat perlindungan *web* pada *login* sistem menggunakan algoritma Rijndael. Dan melindungi web dari serangan SQL *Injection*.

Eka A Dharmawan adalah mahasiswa Program Magister Jurusan Teknik Elektro Universitas Brawijaya, Malang, Indonesia, phone : 085243979979, email : adhitya.wano@gmail.com

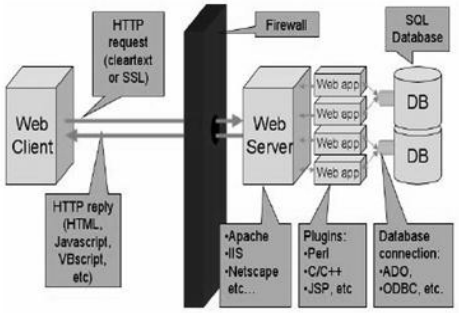
Erni Yudaningtyas adalah Kepala Lab Sistem Kontrol Jurusan Teknik Elektro Universitas Brawijaya, Malang, Indonesia, phone : 08123390449, email : erni_yudaningtyas@yahoo.co.id

M. Sarosa adalah Dosen Jurusan Teknik Elektro, Politeknik Negeri Malang,, Malang, Indonesia, phone : 08122440326, email : msarosa@yahoo.com

II. DASAR TEORI

A. Keamanan Sistem

Masalah keamanan merupakan salah satu aspek penting dari sebuah sistem informasi, sayang sekali masalah keamanan ini seringkali kurang mendapat perhatian dari pemilik dan pengelola sistem infromasi. Jatuhnya informasi ke pihak lain (misal pihak lawan bisnis) dapat menimbulkan kerugian bagi pemilik informasi. Untuk itu keamanan dari sistem informasi yang digunakan harus terjamin dalam batas yang diterima. *Web server* dan *database Server* bagaikan jantung dan otak dari organisme *internet*. Dua komponen ini menjadi komponen pokok dari sebuah aplikasi *internet* yang tangguh dan tepatlah keduanya menjadi target *hacker*. Dalam beberapa kasus kita harus dapat menentukan titik-titik lemah dalam aplikasi tersebut yang bisa menjadi sasaran penyerang. [7]

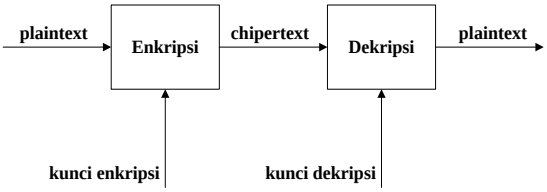


Gambar 1. Contoh Komponen Aplikasi Web[4].

B. Kriptografi

Kriptografi adalah ilmu yang mempelajari mengenai bagaimana cara mengamankan suatu informasi. Pengamanan ini dilakukan dengan mengenkrip informasi tersebut dengan suatu kunci khusus. Informasi ini sebelum dienkrip dinamakan *plaintext*. Setelah dienkrip dengan suatu kunci dinamakan *ciphertext*. Ada empat tujuan mendasar dari ilmu kriptografi ini yang juga merupakan aspek keamanan informasi yaitu :

1. Kerahasiaan, adalah layanan yang digunakan untuk menjaga isi dari informasi dari siapapun kecuali yang memiliki otoritas atau kunci rahasia untuk membuka/mengupas informasi yang telah disandi.
2. Integritas data, adalah berhubungan dengan penjagaan dari perubahan data secara tidak sah. Untuk menjaga integritas data, sistem harus memiliki kemampuan untuk mendeteksi manipulasi data oleh pihak-pihak yang tidak berhak, antara lain penyisipan, penghapusan, dan pensubsitusian data lain kedalam data yang sebenarnya.
3. Autentikasi, adalah berhubungan dengan identifikasi/pengenalan, baik secara kesatuan sistem maupun informasi itu sendiri. Dua pihak yang saling berkomunikasi harus saling memperkenalkan diri. Informasi yang dikirimkan melalui kanal harus diautentikasi keaslian, isi datanya, waktu pengiriman, dan lain-lain.
4. Non-repudiasi., adalah usaha untuk mencegah terjadinya penyangkalan terhadap pengiriman/terciptanya suatu informasi oleh yang mengirimkan/membuat. Istilah-istilah yang digunakan dalam bidang kriptografi :
    a. *Plaintext* (M) adalah pesan yang hendak dikirimkan (berisi data asli).
    b. *Ciphertext* (C) adalah pesan ter-enkrip (tersandi) yang merupakan hasil enkripsi.
    c. Enkripsi (fungsi E) adalah proses pengubahan plaintext menjadi *ciphertext*.
    d. Dekripsi (fungsi D) adalah kebalikan dari enkripsi yakni mengubah *ciphertext* menjadi plaintext, sehingga berupa data awal/asli. [3]



Gambar 2. Diagram Proses Enkripsi Dan Deskripsi. [3]

C. Algoritma Rijndael

Rijndael termasuk dalam jenis algoritma kriptografi yang sifatnya simetri dan *cipher block*. Dengan demikian algoritma ini mempergunakan kunci yang sama saat enkripsi dan dekripsi serta masukan dan keluarannya berupa blok dengan jumlah bit tertentu. Rijndael mendukung berbagai variasi ukuran blok dan kunci yang akan digunakan. Namun Rijndael mempunyai ukuran blok dan kunci yang tetap sebesar 128, 192, 256 bit. Pemilihan ukuran blok data dan kunci akan menentukan jumlah proses yang harus dilalui untuk proses enkripsi dan dekripsi. Berikut adalah perbandingan jumlah proses yang harus dilalui untuk masing-masing masukan.

TABEL I.
JUMLAH PROSES BERDASARKAN BIT BLOK DAN KUNCI. [4]

Panjang kunci (Nk) dalam *words*	Ukuran blok data (Nb) dalam *words*	Jumlah proses(Nr)
4	4	10
6	4	12
8	4	14

Blok-blok data masukan dan kunci dioperasikan dalam bentuk array. Setiap anggota *array* sebelum menghasilkan keluaran *ciphertext* dinamakan dengan state. Setiap *state* akan mengalami proses yang secara garis besar terdiri dari empat tahap yaitu, *AddRoundKey*, *SubBytes*, *ShiftRows*, dan *MixColumns*. Kecuali tahap *MixColumns*, ketiga tahap lainnya akan diulang pada setiap proses sedangkan tahap

MixColumns tidak akan dilakukan pada tahap terakhir. Proses enskripsi adalah kebalikkan dari deskripsi.
Berikut penjelasannya :

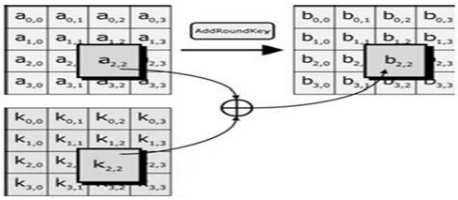
1. *Key Schedule*

Proses key schedule diperlukan untuk mendapatkan subkey-subkey dari kunci utama agar cukup untuk melakukan enkripsi dan dekripsi. Proses ini terdiri dari beberapa operasi, yaitu :

a. Operasi Rotate, yaitu operasi perputaran 8 bit pada 32 bit dari kunci.
b. Operasi SubBytes, pada operasi ini 8 bit dari *subkey* disubstitusikan dengan nilai dari S-Box.
c. Operasi Rcon, operasi ini dapat diterjemahkan sebagai operasi pangkat 2 nilai tertentu dari *user*. Operasi ini menggunakan nilai-nilai dalam Galois field. Nilai-nilai dari Rcon kemudian akan di-XOR dengan hasil operasi *SubBytes*.
d. Operasi XOR dengan w[i-Nk] yaitu *word* yang berada pada Nk sebelumnya.

2. *AddRoundKey*

Pada proses ini *subkey* digabungkan dengan *state*. Proses penggabungan ini menggunakan operasi XOR untuk setiap byte dari subkey dengan *byte* yang bersangkutan dari *state*. Untuk setiap tahap, *subkey* dibangkitkan dari kunci utama dengan menggunakan proses *key schedule*. Setiap *subkey* berukuran sama dengan *state* yang bersangkutan. Proses *AddRoundKey* diperlihatkan pada Gambar 3.



Gambar 3. Proses AddRoundKey. [4]

3. *SubBytes*

Rijndael hanya memiliki satu S-box. Kriteria desain untuk kotak S yang dibuat sedemikian rupa sehingga tahan terhadap diferensial linear yang dikenal sebagai pembacaan sandi dan menyerang menggunakan manipulasi aljabar. Koordinat x merupakan digit pertama sedangkan y yang kedua dari bilangan hexadesimal, dapat dilihat pada Tabel II.

Contoh :

$$S = \begin{matrix} 19 & A0 & 9A & E9 \\ 3D & F4 & C6 & F8 \\ E3 & E2 & 8D & 48 \\ BE & 2B & 2A & 08 \end{matrix} \quad (1)$$

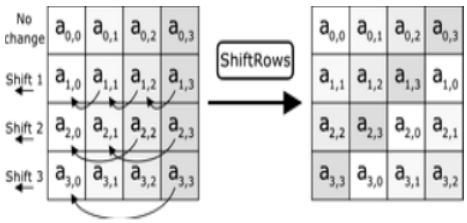
Sehingga mendapatkan S' sebagai berikut

$$S' = \begin{matrix} D4 & E0 & B8 & 1E \\ 27 & BF & B4 & 41 \\ 11 & 98 & 5D & 52 \\ AE & F1 & E5 & 30 \end{matrix} \quad (2)$$

Proses sub *bytes* ditampilkan dalam perpindahan *state* diatas.

4. *Shift Rows*

Proses *Shift Rows* akan beroperasi pada tiap baris dari tabel state. Proses ini akan bekerja dengan cara memutar byte-byte pada 3 baris terakhir (baris 1, 2, dan 3) dengan jumlah perputaran yang berbeda-beda. Baris 1 akan diputar sebanyak 1 kali, baris 2 akan diputar sebanyak 2 kali, dan baris 3 akan diputar sebanyak 3 kali. Sedangkan baris 0 tidak akan diputar.



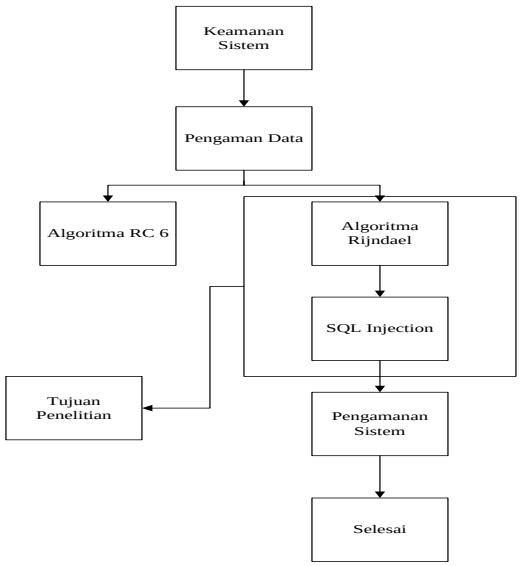
Gambar 4. Proses Shift Rows. [8]

5. *MixColums*

Proses *MixColumns* akan beroperasi pada tiap kolom dari tabel state. Operasi ini menggabungkan 4 bytes dari setiap kolom tabel state dan menggunakan transformasi linier Operasi *Mix Columns* memperlakukan setiap kolom sebagai polinomial 4 suku dalam Galois field dan kemudian dikalikan dengan c(x) modulo (x4+1), dimana c(x)=3x3+x2+x+2. Kebalikkan dari polinomial ini adalah c(x)=11x3+13x2+9x+14. Operasi *MixColumns* juga dapat dipandang sebagai perkalian matrix.

III. KERANGKA KONSEP PENELITIAN

A. Kerangka Berfikir



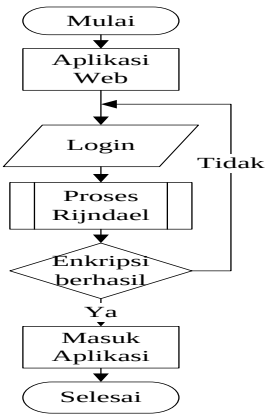
Gambar 5. Kerangka Berfikir

Pengamanan pengiriman data, difokuskan pada algoritma Rijndael dan *database* yang dilindungi dari SQL *Injection*. Pengiriman data dimulai pada saat *login* sistem dengan melakukan autentikasi terhadap *login*. Algoritma Rijndael lebih dipilih dari RC 6, karena Rijndael memiliki fleksibilitas yang tinggi dan dapat diterapkan pada *platform* yang beragam sementara RC6 kurang memiliki kebebasan di berbagai platform. RC6

merupakan algoritma yang merupakan keturunan dari RC5 yang juga merupakan kandidat AES (*Advanced Encryption Standard*). Pada mulanya, perancangan RC6 diawali ketika RC5 dianggap dapat dijadikan kandidat untuk mengikuti kompetisi pemilihan AES. Algoritma Rijndael dan SQL *Injection* menjadi tujuan utama dalam penelitian ini. Serta menjadi pengamanan sistem informasi remunerasi berbasis *web*. Kerangka berfikir dalam penelitian ini terdapat dalam Gambar 5.

B. Flowchart dan Solusi Masalah

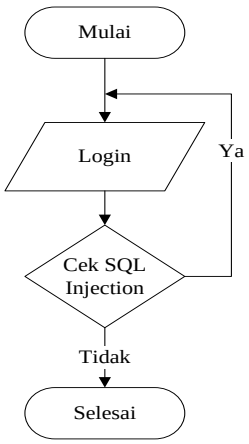
Flowchart dan solusi masalah dalam penelitian ini terdapat dalam Gambar 6.



Gambar 6. Kerangka Berfikir

Dalam *flowchart* solusi masalah, aplikasi menggunakan *security* algoritma Rijndael. Tampak dalam Gambar 6 dimulai dari masuk aplikasi, kemudian input pada *login*, setelah dimasukkan nama dan *password* maka akan dienkripsi,dan ketika enkripsi berhasil, dan data valid, maka aplikasi dapat digunakan.

C. Model Analisis Simulasi Serangan



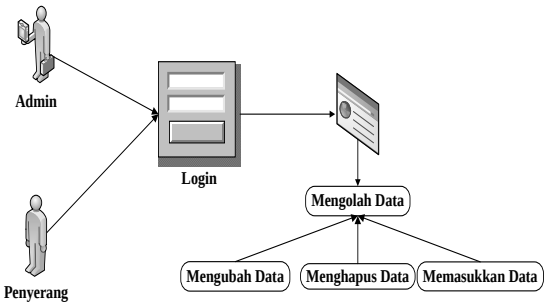
Gambar 7. *Flowchart* Pengecekan *Login*.

Flowchart dibawah ini merupakan pecahan dari *flowchart* solusi masalah pada Gambar 6, pengecekan *login* dilakukan pada *form input login*. Dalam *flowchart* simulasi serangan menggunakan SQL *Injection*, akan terus di masukkan kode serangan dalam *form login* untuk mencoba sistem pengamanan apakah berhasil atau tidak. SQL *Injection* dicoba pada *form login* jika kondisinya 'ya' akan berulang ke posisi form login, jika tidak berhasil maka berhak masuk ke input login sama dengan *database*, kemudian dicocokkan, kalo cocok akan masuk ke aplikasi dan sebagai bukti bahwa SQL *Injection* tidak dapat meretas sistem.

IV. HASIL DAN PEMBAHASAN

A. Perancangan Antarmuka

Dalam perancangan antarmuka dituntut untuk membuat antarmuka yang mudah dimengerti oleh pengguna sehingga aplikasi akan lebih interaktif dapat dilihat dalam Gambar 8.



Gambar 8. Perancangan Antarmuka

1. Definisi Aktor

Tabel III dibawah ini menjelaskan tentang definisi setiap Aktor yang ada pada sistem yaitu Admin dan penyerang.

TABEL III.
DEFINISI AKTOR

No	Aktor	Deskripsi
1	Admin	User yang berhak mengakses dan melakukan pengolahan data Admin
2	Penyerang	User yang tidak sah (penyerang) yang dapat masuk ke sistem Admin dan memanipulasi data pada sistem seperti mengubah data memasukkan data maupun menghapus data

2. Definisi Alur Sistem

TABEL IV.
DEFINISI ALUR SISTEM

No	Proses	Deskripsi
1	Login	Login merupakan proses pengecekan hak akses siapa saja yang dapat masuk ke sistem. Dalam hal ini hanya Admin yang berhak login dan melakukan proses pengolahan data.
2	Mengolah Data	Merupakan proses mengolah data seperti memasukkan data, merubah data, dan menghapus data.
3	Memasukkan Data	Merupakan proses memasukkan data informasi ke dalam database
4	Mengubah Data	Merupakan proses perubahan data informasi yang ada didalam database
5	Menghapus Data	Merupakan proses menghapus data informasi yang ada didalam database

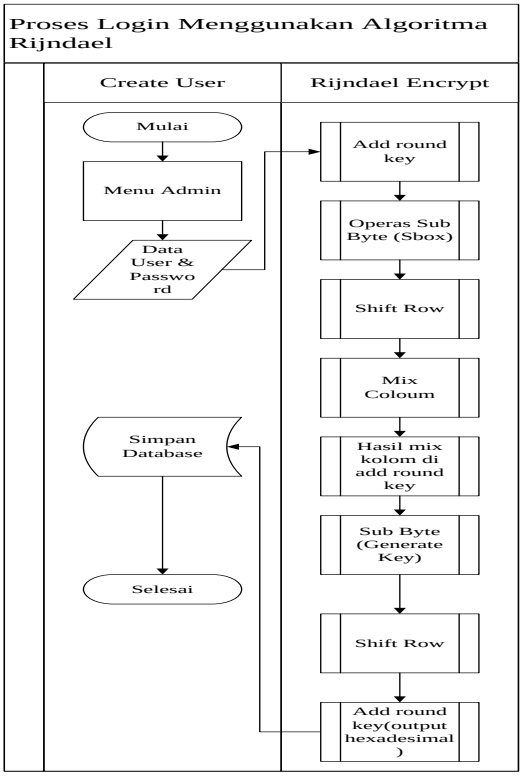
Tabel IV dibawah ini menjelaskan tentang definisi setiap proses yang ada pada sistem yaitu *login*, Mengolah data, Memasukkan data, Mengubah data

dan Menghapus data.

B. Analisis Cara Kerja Algoritma Rijndael

- Proses Enkripsi

  Pada system flow enkripsi dalam algoritma Rijndael, mempunyai beberapa urutan proses terlihat pada Gambar 8.



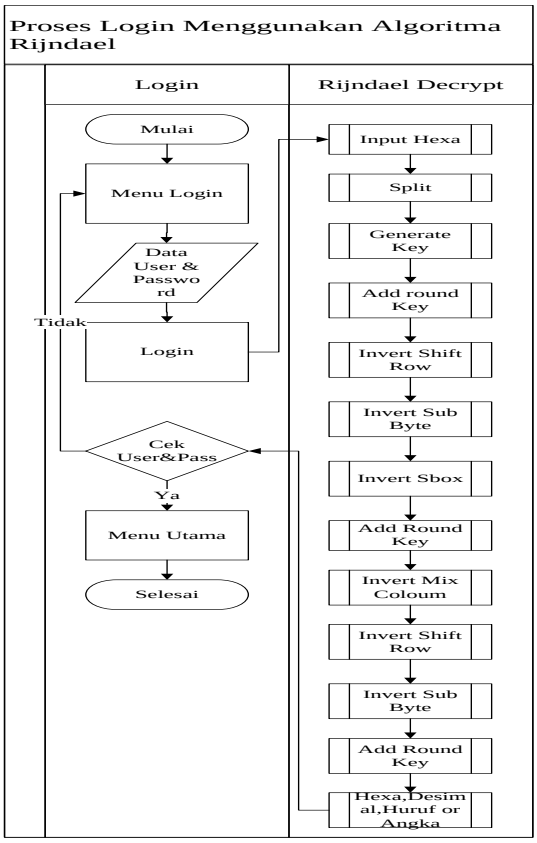
Gambar 8. *System flow* Proses Enkripsi

  - Pada saat membuat user dan password baru, data akan di enkrip.
  - Proses enkrip dimulai dengan add_round_key (kalimat+kunci).
  - Kemudian membuat operasi sub byte (dari Sbox).
  - Shift row dengan cara memindahkan baris-baris dalam kolom.
  - Mix coloum (mix matriks) dalam bentuk bit.
  - Hasil mix kolom pertama menghasilkan schedule kemudian di add_round_key.
  - Langkah *add round key – mix* coloum diproses sebanyak 9 kali putaran.
  - Sub byte dengan membangkitkan kunci yang selanjutnya.
  - Shift row
  - Add_round_key tanpa mix coloum keluarannya hexadecimal.
  - Setelah proses enkripsi berhasil maka data user dan password akan disimpan ke database untuk dilakukan pemanggilan pada saat proses dekripsi.

- Proses Dekripsi

Pada *system flow* dekripsi dalam algoritma Rijndael, mempunyai beberapa urutan proses terlihat pada Gambar 9.

  - Menu login ditampilkan, kemudian input user dan password.
  - Data user dan password akan dicocokkan dengan database.
  - Kemudian dilanjutkan dengan tahapan dekripsi yang dimulai dengan masukan hexadecimal 32 bit.
  - Dibagi 2 menjadi 16 + 16 (split)



Gambar 9. *System flow* Proses Dekripsi

  - Membangkitkan key pertama jadi key yang terakhir.
  - Add_round_key terakhir.
  - Invert shift row.
  - Invert sub byte.
  - Invert Sbox.
  - Add_round_key.
  - Invert mix coloum.
  - Invert shift row.
  - Invert sub byte.
  - Langkah *invert* diproses sebanyak 9 kali putaran.
  - Add_round_key.
  - Keluaran hexa, diubah ke desimal kemudian diubah lagi ke huruf atau angka.

C. Pengujian Algoritma Rijndael

Pada hasil dan pengujian penggunaan dalam melindungi login sistem menggunakan algoritma Rijndael, hasil sistem berisi antarmuka program dan pengujian sistem yang telah dibuat. Hasil pengujian sesuai dengan proses runtunan jalannya algoritma. Pada proses pengujian, terdapat dua tahap, yaitu pengujian SQL *Injection* tanpa Rijndael dan pengujian SQL *Injection* dengan menggunakan Rijndael pada *login*

sistem.

- Pengujian Serangan SQL Injection Pada Web tanpa Rijndael.

Dalam mengimplementasikan serangan SQL *Injection* ini dibutuhkan *website* yang memiliki keamanan yang rendah. Website dengan keamanan yang rendah memudahkan bagi penyerang untuk melakukan serangan SQL Injection. Berikut beberapa skenario yang akan diuji dalam penelitian ini dalam Tabel V.

TABEL V.
SERANGAN SQL INJECTION PADA WEB TANPA RIJNDAEL

No	Variabel SQL *Injection*	Dampak Serangan
1	'or 1=1#	Berhasil Login
2	“or 1=1-	Gagal Login
3	‘or 1=1-	Gagal Login
4	‘or”a’=’a’#	Gagal Login
5	'or"a"="a"#	Berhasil Login
6	Admin’ #	Gagal Login
7	' or 'x'='x'#	Berhasil Login
8	Admin’ or 1=1 #	Gagal Login
9	hi' or 'a'='a'#	Berhasil Login
10	hi”) or (“a”=”a	Gagal Login
11	Admin' or 0=0#	Berhasil Login
12	“or ”a”=”a	Gagal Login
13	admin' OR '1'='1	Berhasil Login
14	' or 0=0 #	Berhasil Login
15	' or 0=0 --	Gagal Login
16	" or 0=0 --	Gagal Login
17	" or 0=0 #	Gagal Login
18	admin ' or 'x'='x	Berhasil Login
19	" or "x"="x	Gagal Login
20	‘) or ('x'='x	Gagal Login
21	' or 1=1--	Gagal Login
22	" or 1=1--	Gagal Login
23	or 1=1--	Gagal Login
24	' or a=a--	Gagal Login
25	" or "a"=”a	Gagal Login
26	hi” or 1=1 –	Gagal Login
27	admin’–	Gagal Login
28	’having 1=1–	Gagal Login
29	hi' or 'a'='a	Gagal Login
30	hi" or 1=1 --	Gagal Login
31	“or 0=0 –	Gagal Login
32	admin 'or"a"="a"#	Berhasil Login
33	hi" or "a"="a	Gagal Login
34	hi' or 1=1 --	Gagal Login
35	hi' or 'a'='a#	Gagal Login
36	hi') or ('a'='a	Gagal Login
37	hi") or ("a"="a#	Gagal Login
38	admin hi' or a'='a'#	Gagal Login
39	admin'or 1=1#	Berhasil Login
40	admin hi' or 'a'='a'#	Berhasil Login

Berdasarkan data percobaan dalam Tabel V. Dapat disimpulkan bahwa *web* tanpa Rijndael kurang efektif menangani serangan SQL *Injection* dan banyak yang berhasil meretas sistem.

Gambar 10. Tampilan Pengujian Memakai SQL *Injection*

Pada Gambar 10. menjelaskan bahwa halaman *login* pada *web* di retas dengan cara memasukkan kode SQL *Injection*. Kode SQL *Injection*, akan dimasukkan pada bagian *username*. SQL *Injection* mampu meretas sistem *web*, meskipun terlindungi dengan adanya *user* dan *password*.

Pada Gambar 11. menjelaskan, web berhasil diretas dengan memasukkan kode SQL *Injection*. Dan berhasil melakukan proses *login* tanpa menggunakan *username* ataupun *password*.

- Pengujian Serangan SQL *Injection* Pada Web dengan Rijndael.

TABEL VI.
SERANGAN SQL INJECTION PADA WEB DENGAN RIJNDAEL

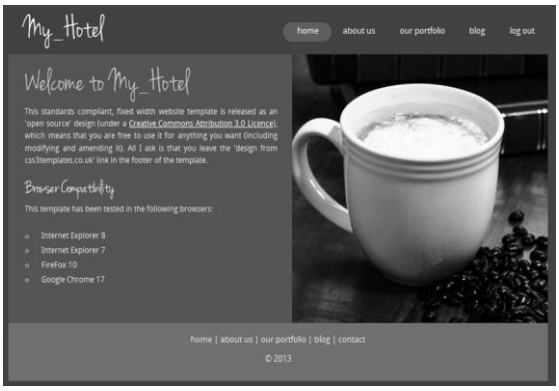
No	Variabel SQL *Injection*	Dampak Serangan
1	'or 1=1#	Gagal Login
2	“or 1=1-	Gagal Login
3	‘or 1=1-	Gagal Login
4	‘or”a’=’a’#	Gagal Login
5	'or"a"="a"#	Gagal Login
6	Admin’ #	Gagal Login
7	' or 'x'='x'#	Gagal Login
8	Admin’ or 1=1 #	Gagal Login
9	hi' or 'a'='a'#	Gagal Login
10	hi”) or (“a”=”a	Gagal Login
11	Admin' or 0=0#	Gagal Login
12	“or ”a”=”a	Gagal Login
17	" or 0=0 #	Gagal Login
18	admin ' or 'x'='x	Gagal Login
19	" or "x"="x	Gagal Login
20	‘) or ('x'='x	Gagal Login
21	' or 1=1--	Gagal Login
22	" or 1=1--	Gagal Login
23	or 1=1--	Gagal Login
24	' or a=a--	Gagal Login
25	" or "a"=”a	Gagal Login
26	hi” or 1=1 –	Gagal Login
27	admin’–	Gagal Login
28	’having 1=1–	Gagal Login
29	hi' or 'a'='a	Gagal Login
30	hi" or 1=1 --	Gagal Login
31	“or 0=0 –	Gagal Login
32	admin 'or"a"="a"#	Gagal Login
33	hi" or "a"="a	Gagal Login
34	hi' or 1=1 --	Gagal Login
35	hi' or 'a'='a#	Gagal Login
36	hi') or ('a'='a	Gagal Login
37	hi") or ("a"="a#	Gagal Login
38	admin hi' or a'='a'#	Gagal Login
39	admin'or 1=1#	Gagal Login
40	admin hi' or 'a'='a'#	Gagal Login

Pada Tabel VI. menjelaskan, ketika diberikan kode SQL *Injection*, *web* tidak berhasil diretas, dan dinyatakan *login* gagal. Membuktikan bahwa Rijndael mampu memberikan pengamanan yang baik terhadap sebuah *web*. Pada Tabel VI. membuktikan bahwa serangan SQL *Injection* tidak berhasil pada *web* dengan keamanan yang baik menggunakan Rijndael.

Pada Gambar 12. *form login* diberikan kode SQL *Injection*. Web akan diretas dengan cara memberikan kode SQL *Injection*.

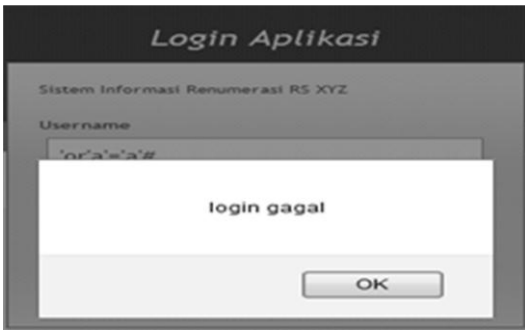
Pada Gambar 13. menjelaskan, ketika diberikan kode SQL *Injection web* tidak berhasil diretas, dan dinyatakan *login* gagal. Membuktikan bahwa Rijndael mampu memberikan pengamanan yang baik terhadap

sebuah *web*.



Gambar 11. Tampilan Web Yang Berhasil Diretas

Gambar 12. Tampilan Pengujian SQL Injection Pada Rijndael



Gambar 13. Tampilan Web Gagal Diretas

V. KESIMPULAN DAN SARAN

A. Kesimpulan

Dari hasil perancangan sistem kemudian dilanjutkan dengan pengambilan data, pengujian dan analisa, maka dapat disimpulkan sebagai berikut :

1. Dari pengujian sebanyak 40 kode SQL *Injection*, 11 berhasil masuk kedalam sistem tanpa Rinjdael. Setelah dimasukkan Rijndael ke-40 skenario SQL *Injection* gagal.
2. Algoritma Rijndael dapat digunakan dalam memproteksi SQL *Injection* dimana SQL *Injection* menyerang *database server*. Karena *database server* merupakan tempat penyimpanan data yang harus dijaga dan dlindungi dari pihak yang tidak berkepentingan untuk mengolah data.

B. Saran

Berikut beberapa saran dalam penelitian ini untuk diteliti lebih lanjut :

1. Algoritma Rijndael dapat digabungkan dengan beberapa metode pengamanan lain untuk melindungi *web* sistem dari para penyerang.
2. Rijndael hanya memproteksi *database*. Tetapi dari segi yang lain perlu ditambahkan algoritma keamanan untuk melindungi *router* atau jaringan.

DAFTAR PUSTAKA

[1] Rahardjo B. 1999. Keamanan Sistem Informasi Berbasis Internet. PT Insan Komunikasi. Bandung.

[2] Patel, N., Mohammed, F., dan Soni. S. 2011. SQL Injection Attacks: Techniques and Protection Mechanisms. IJCSE.

[3] Fairuzabadi, M. 2010. Implementasi Kriptografi Klasik Menggunakan Borland Delphi. Jurnal Dinamika Informatika, Vol 4.

[4] Surian, D. 2006. Algoritma Kriptografi AES Rijndael : TESLA, Vol 8.

[5] Soyjaudah, K.M.S., Hosany. M. A., dan Jamaloodeen, A. 2004. Design and Implementation of Rijndael algorithm for GSM Encryption. IEEE.

[6] Majumder, J., Saha, G. 2009. Analysis of SQL Injection Attack. Special Issue of International Journal of Computer Science & Informatics (IJCSI), ISSN (PRINT): 2231–5292, Vol.- II, Issue-1, 2.

[7] Stiawan, D. 2005. Sistem Keamanan Komputer, Elex Media Komputindo.

[8] Jamil, T. 2004, The Rijndael algorithm, Potentials, IEEE, Vol 23.

[9] Cartrysse, K., J.C.A. van der Lubbe. 2004. The Advanced Encryption Standard: Rijndael : Supplement to the books ”Basic methods of cryptography”and”Basismethodencryptografie”http://mail.vssd.nl/hlf/e012rijndael.pdf. diakses tanggal 9 Mei 2013

[10] Fatta, H. 2007. Analisis dan Perancangan Sistem Informasi untuk Keunggulan Bersaing Perusahaan dan Organisasi Modern. Andi Offset, Yogyakarta.

[11] Griffin, Ricky. E. 1999. Management. Edisi kelima, New Jersey.

[12] Herlambang, S dan Tanuwijaya, H. 2005. Sistem Informasi: konsep, teknologi, dan manajemen. Graha ilmu, Yogyakarta.

[13] Budhi, S., G., Liem. R., dan Surya. D. 2010. Kombinasi Metode Steganografi Parity Coding Dan Metode Enkripsi AES Rijndael Untuk Pengamanan Dokumen Elektronik. Jurnal Informatika, Vol 9.

[14] Igor, Beny. 2009. Perbandingan Algoritma RC6 dengan Rijndael pada AES. http://informatika.stei.itb.ac.id/~rinaldi.munir/Kriptografi/2007-2008/Makalah1/MakalahIF5054-2007-A-038.pdf. diakses tanggal 10 Mei 2013

[15] Information-Technology Promotion Agency, Japan (IPA). 2011. How to Secure Your Website, five edition.

[16] Jogiyanto, H.M. 2005. Analisis Desain dan Desain Sistem Informasi. Elex Media Komputerindo. Jakarta.

[17] Kamus Besar Bahasa Indonesia 2008.

[18] Kendall, K.E., Kendall J.E. 2003. Analisis dan Perancangan Sistem Jilid 1. Prehallindo. Jakarta.

[19] Sirait, Justine. T. 2007. Memahami Aspek-Aspek Pengelolaan Sumber Daya Manusia dalam Organisasi. Grasindo. Jakarta.

[20] Soendoro, H., Haryanto. T. 2005. Sistem Informasi: konsep, teknologi, dan manajemen. Graha ilmu, Yogyakarta

[21] Surya, M. 2004. Bunga Rampai Guru dan Pendidikan. Edisi Pertama, PT Balai Pustaka. Jakarta.

[22] Sutabri, T. 2004. Analisa Sistem Informasi. Andi Offset. Yogyakarta.

[23] Wahyono, T. 2004. Sistem Informasi Konsep Dasar, Analisis Desain dan Implementasi. Graha Ilmu. Klaten.

[24] Wali, M.F., Rehan. M. 2005. Effective Coding and Performance Evaluation of the Rijndael Algorithm (AES), Engineering Sciences and Technology. SCONEST.

[25] Widiasari, R.I. 2012. Combining Advanced Encryption Standard (AES) and One Time Pad (OTP) Encryption for Data Security. International Journal of Computer Applications, Volume 57.

TABEL II.

S-BOX. [8]

		Y															
		0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**A**	**B**	**C**	**D**	**E**	**F**
	0	63	7C	77	7B	F2	6B	6F	C5	30	O1	67	2B	FE	D7	AB	76
	1	CA	82	C9	7D	FA	59	47	F0	AD	D4	A2	AF	9C	A4	72	CO
	2	B7	FD	93	26	36	3F	F7	CC	34	A5	E5	F1	71	D8	31	15
	3	O4	C7	23	C3	18	96	O5	9A	O7	12	80	E2	EB	27	B2	75
	4	O9	83	2C	1A	18	6E	5A	A0	52	3B	D6	B3	29	E3	2F	84
	5	53	D1	OD	ED	20	FC	B1	58	6A	CB	BE	39	4A	4C	58	CF
	6	DD	EF	AA	FB	43	4D	33	85	45	F9	O2	7F	50	3C	9F	A8
X	**7**	51	A3	4O	8F	92	9D	38	F5	BC	B6	DA	21	10	FF	F3	D2
	8	CD	OC	13	EC	5F	97	44	17	C4	A7	7E	3D	64	5D	19	73
	9	6O	81	4F	DC	22	2A	90	88	46	EE	B8	14	DE	5E	OB	DB
	A	EO	32	3A	OA	49	O6	24	5C	C2	D3	AC	62	91	95	E4	79
	B	E7	C8	37	6D	8D	D5	4E	A9	6C	56	F4	EA	65	7A	AE	O8
	C	BA	78	25	2E	1C	A6	B4	C6	E8	DD	74	1F	4B	BD	88	8A
	D	70	3E	B5	66	48	03	F6	OE	61	35	57	B9	86	C1	1D	9E
	E	E1	FB	98	11	69	D9	8E	94	9B	1E	87	E9	CE	55	28	DF
	F	8C	A1	89	OD	BF	E6	42	68	41	99	2D	OF	BO	54	BB	16