# PENGAMANAN DATA USER PADA DATABASE MENGGUNAKAN KRIPTOGRAFI RIVEST SHAMIR ADLEMAN DAN CIPHER BLOCK CHAINING

**Ipan Tropiana[1], Gunawan Abdillah[2], Ridwan Ilyas[3]**
[1,2,3]*Informatika, Fakultas Sains dan Informatika, Universitas Jenderal Achmad Yani Cimahi*
*Jalan Terusan Jendral Sudirman, Cimahi, Jawa Barat, 40285*
*E-mail: ipantrp@gmail.com, abi_zakiyy@yahoo.com, ilyas@lecture.unjani.ac.id*

**ABSTRAKS**
*Data yang tersimpan di dalam database tidak sepenuhnya aman, ada beberapa teknik yang sering digunakan untuk mencuri data dari database, misalnya teknik SQL Injection. Dengan menggunakan salah satu teknik SQL Injection yaitu memanfaatkan bug yang ada pada URL, pencuri memungkinkan untuk melihat isi dari database, jika tidak dilakukan pengamanan maka data tersebut dapat bebas dilihat oleh orang tersebut. Penelitian ini telah membangun sistem pengamanan terhadap data user dengan metode Rivest Shamir Adleman (RSA) dan Cipher Block Chaining (CBC). Pengamanan ini memungkinkan isi data terenkripsi sehingga pada saat sistem diretas dengan SQL Injection maka data tidak dapat dipahami, data yang telah dienkripsi dapat didekripsi kembali untuk menjaga keaslian data. Hasil dari pengujian yang telah dilakukan, sistem ini berhasil melakukan enkripsi pada data tabel sehingga data menjadi karakter acak yang tidak dapat dipahami, pada saat database diretas dengan SQL Injection pun data yang tampil adalah data yang telah terenkripsi.*

*Kata Kunci: Database, Rivest Shamir Adleman, Cipher Block Chaining, Enkripsi, Dekripsi*

## 1. PENDAHULUAN

### 1.1 Latar Belakang Masalah

Data pada *database* tidak sepenuhnya aman dari pencurian, banyak terdapat celah yang dapat digunakan untuk memanipulasi hak akses pada *database*, di antaranya adalah teknik SQL Injection. Contoh serangan yang dapat dilakukan dengan SQL Injection adalah melalui URL, sebuah bahasa pemrograman seperti PHP mengakses *database* melalui SQL *query*. Jika data yang dikirim langsung ke *database* dan tidak disaring dengan benar, maka penyerang dapat menyisipkan perintah SQL nya sebagai bagian dari *input.*

Terdapat banyak sekali teknik pengamanan data yang dapat digunakan, salah satunya adalah dengan menggunakan enkripsi atau mengubah teks yang biasa (*plain text*) menjadi teks yang tidak mudah untuk dibaca (*cipher text*) atau sering dikenal sebagai teknik kriptografi. Dengan menggunakan pengamanan ini, setidaknya dapat mempersulit para *hacker* untuk dapat mencuri informasi penting yang ada. Terdapat berbagai macam teknik kriptografi yaitu simetris, asimetris, dan ada juga yang menggabungkan kedua teknik tersebut karena kriptografi simetris dan asimetris memiliki kelemahan masing–masing(Basri, 2016).

Teknik asimetris melibatkan dua kunci dalam prosesnya yaitu kunci *public* yang dapat diketahui oleh orang lain dan kunci *private* yang hanya dia sendiri yang boleh mengetahuinya. Sedangkan teknik simetris hanya menggunakan satu buah kunci untuk prosesnya, keuntungan menggunakan satu kunci adalah proses enkripsi berjalan lebih cepat meskipun menggunakan data yang relatif banyak. Teknik simetris dapat digunakan untuk mengamankan data karena cepat meskipun memroses banyak data sedangkan teknik asimetris dapat digunakan untuk otentikasi kunci enkripsi karena ukuran kunci enkripsi tidak terlalu besar(Lal, 2017)(Gupta & Sharma, 2012). Kunci adalah hal yang sangat penting dalam kriptografi, menjaga agar kunci tetap aman akan membuat data sulit untuk diakses orang lain, kecepatan proses enkripsi atau dekripsi juga adalah salah satu faktor yang menentukan efisien atau tidaknya metode yang digunakan(Menon, Joy, Emmanuel, & Paul, 2017)(Agrawal & Sharma, 2014).

Pada penelitian terdahulu dijelaskan salah satu metode kriptografi yaitu Cipher Block Chaining (CBC) adalah salah satu metode enkripsi yang memanfaatkan *feedback* dari karakter yang telah dienkripsi, artinya karakter yang telah dienkripsi tersebut menentukan enkripsi selanjutnya. *Feedback* didapatkan dari kunci tambahan yaitu Initialization Vector (IV), konsep nilai IV ini adalah memperbaharui dirinya sendiri sesuai dengan hasil proses sebelumnya, maka setiap hasil enkripsi tidak memiliki jaminan hasilnya akan sama karena setiap karakter hasil enkripsi akan mempengaruhi hasil enkripsi karakter lainnya(Lestiawan & Purnama, 2016). Penelitian yang lain menjelaskan metode RSA, di mana metode ini adalah salah satu metode kriptografi asimetris terbaik(Arya, Aswal, & Kumar, 2012). RSA menggunakan dua kunci *public* dan satu kunci *private*, proses enkripsi pada RSA menggunkan kunci *private* dan satu kunci *public*, sedangkan untuk dekripsinya RSA menggunakan dua kunci *public*.

Penggunaan enkripsi dilakukan agar data yang telah diamankan dapat dikembalikan menjadi data asli melalui proses dekripsi, berbeda dengan teknik *hashing* yang merupakan teknik pengamanan satu arah yang

mengakibatkan data asli tidak dapat dikemballikan kecuali dengan mekanisme tertentu. Ada beberapa faktor yang mempengaruhi waktu proses enkripsi ataupun dekripsi yaitu besar *file* yang akan dienkripsi, semakin besar ukuran *file* yang akan dienkripsi maka akan semakin lama proses enkripsi dan sebaliknya semakin kecil ukuran *file* maka semakin cepat proses enkripsi(Henry, Kridalaksana, & Arifin, 2016). Karena teknik asimetris dan simetris memiliki kelemahan dan kelebihan masing-masing, meskipun teknik asimetris lebih baik daripada simetris soal keamanan tetapi tetap saja teknik asimetris memiliki kelemahan. Maka penggabungan metode simetris dan asimetris dapat menjadi solusi untuk menjadikan pengamanan lebih kuat.

Data *user* sering menjadi sasaran untuk diretas karena dengan memilikinya maka peretas akan mendapatkan hak akses untuk masuk ke sistem, oleh karena itu pengamanan data diperlukan untuk mencegah terjadinya pencurian. Dengan melakukan enkripsi pada data, meskipun peretas mampu melakukan teknik SQL Injection dan berhasil mengakses isi dari data tetapi data tidak akan dapat dibaca.

## 1.2 Tinjauan Pustaka

### 1.2.1 Kriptografi

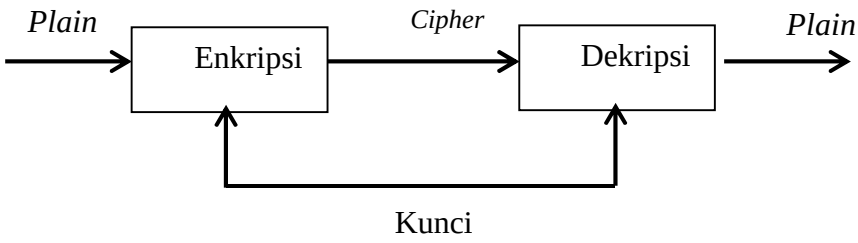
Keamanan adalah salah satu hal yang diinginkan oleh setiap orang, termasuk keamanan dalam penyimpanan data. Untuk mewujudkan keamanan tersebut, salah satu teknik yang dapat dilakukan yaitu Kriptografi, di mana teknik ini memiliki aspek seperti kerahasiaan, otentikasi, integritas, dan nirpenyangkalan. Dengan aspek–aspek tersebut maka diharapkan setiap orang yang ingin bertukar informasi dapat terjamin keamanannya.

Pada penelitian sebelumnya yang menganalisa salah satu algoritma asimetris yaitu RSA, di mana pada penelitian tersebut dijelaskan bahwa teknik asimetris melakukan penggabungan secara kriptografi dua buah kunci yang berhubungan yaitu kunci *public* dan kunci *private*(Chandra, 2016). Kedua kunci tersebut dibuat pada waktu yang bersamaan dan berhubungan secara matematis. Secara matematis, kunci *private* dibutuhkan untuk melakukan operasi *invers* terhadap kunci *public* dan kunci *public* dibutuhkan untuk melakukan operasi *invers* terhadap operasi yang dilakukan oleh kunci *private*. Teknik asimetris ini memiliki tingkat keamanan yang tinggi karena penggunaan dua kunci yang saling berhubungan untuk enkripsi maupun dekripsi. Namun dari segi kecepatan, teknik ini relatif lebih lambat dari teknik simetris jika data yang diamankan cukup banyak(Tripathi & Sanjay, 2014).

Pada penelitian sebelumnya yang membahas tentang penggabungan metode kriptografi yaitu Cipher Block Chaining dan Triangle Chain Cipher, di mana ke dua metode tersebut adalah metode simetris yang menggunakan satu kunci. Tetapi dengan menggunakan dua metode, akan lebih memperkuat pengamanan karena enkripsi yang dilakukan menjadi dua kali. Penelitian ini mengamankan data *login* yaitu *password*, enkripsi dilakukan dengan metode Cipher Block Chaining terlebih dahulu kemudian hasil *cipher text* dienkripsi kembali oleh metode Triangle Chain Cipher, hasil dari enkripsi inilah yang akan disimpan di dalam *database*(Lombu, Tarihoran, & Gulo, 2018). Pada penelitian lain membahas tentang modifikasi algoritma Cipher Block Chaining di mana biasanya kunci untuk enkripsi bernilai sama setiap blok, dengan modifikasi ini membuat kunci akan berubah mengikuti panjang kunci setiap blok. Modifikasi ini adalah penggabungan dengan metode Vignere Cipher, dengan demikian data akan lebih aman dari peretasan(Rochmah & Ardiansyah, 2013).

### 1.2.2 Kriptografi Simetris

Kriptografi simetris adalah kriptografi yang menggunakan kunci yang sama untuk proses enkripsi maupun dekripsinya. Karena hanya menggunakan satu kunci, pemrosesan enkripsi dan dekripsi dengan kriptografi simetris ini relatif cepat dibandingkan dengan asimetris(Yassein, Aljawarneh, Qawasmeh, Mardini, & Khamayseh, 2017).

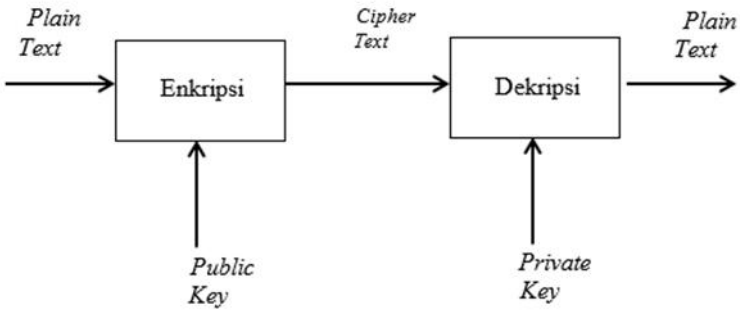



**Gambar 1. Proses kriptografi simetris**

Gambar 1 menunjukan proses dari kriptografi simetris, terlihat bahwa teknik ini menggunakan kunci yang sama untuk proses enkripsi maupun dekripsinya, kunci tersebut haruslah rahasia karena jika kunci itu didapatkan oleh orang lain maka bukan tidak mungkin data yang telah diamankan dapat dicuri dengan mudah.

### 1.2.3 Kriptografi Asimetris

Kriptografi asimetris termasuk dalam teknik kriptografi yang menggunakan dua kunci yaitu kunci *public* dan kunci *private* untuk pemrosesannya. Kunci *public* adalah kunci yang dapat diketahui oleh orang karena kunci tersebut tidak dapat dipakai untuk mengakses data, sedangkan kunci *private* merupakan kunci yang hanya dapat diketahui oleh pihak tertentu saja, skema proses kriptografi asimetris dapat dilihat pada Gambar 2.

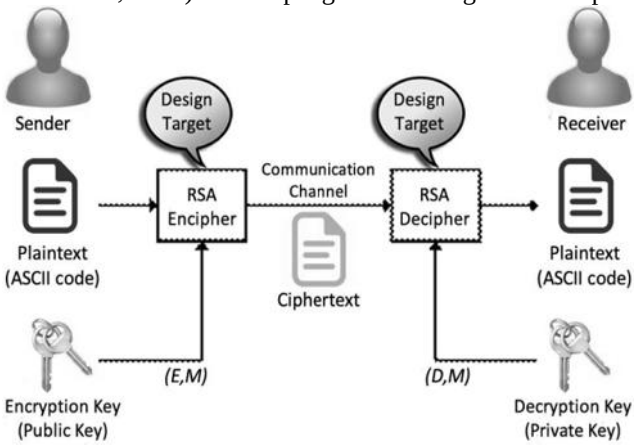



**Gambar 2. Proses kriptografi asimetris**

Gambar 2 menunjukan proses enkripsi dan dekripsi dengan teknik simetris, terlihat bahwa saat melakukan proses enkripsi kunci yang digunakan adalah kunci *public* sedangkan kunci *private* digunakan untuk proses dekripsi. Teknik asimetris ini memiliki kelemahan yaitu proses yang cukup memakan waktu dalam enkripsi maupun dekripsi untuk jumlah data yang besar karena penggunaan dua kunci, untuk keunggulannya teknik ini dapat melakukan pengamanan ekstra pada data dengan kunci *public* dan *private*(Wulansari, Muslim, & Sugiharti, 2016).

### 1.2.4 Rivest Shamir Adleman (RSA)

RSA merupakan salah satu metode kriptografi yang menggunakan kunci *private* dan kunci *public* untuk proses enkripsi dan dekripsi atau dalam istilahnya sering disebut teknik asimetris, algoritma ini menggunakan kunci *private* untuk melakukan enkripsi dan untuk dekripsinya menggunakan kunci *public*. Pada saat ini algoritma RSA banyak digunakan pada *browser* dan *web* server untuk pengamanan komunikasi(Galla, Koganti, & Nuthalapati, 2017), namun algoritma ini masih dapat menerima serangan salah satunya adalah serangan Brute Force Attack(Shende, Sudi, & Kulkarni, 2017). Proses pengamanan dengan RSA dapat dilihat pada Gambar 3.




**Gambar 1. Skema algoritma RSA**

Sebelum data dienkripsi, kunci diperlukan untuk melakukan proses pembangkitan kunci. Tahapan pembuatan kunci RSA adalah sebagai berikut:

a. *User* menentukan dua buah bilangan P dan Q yang merupakan bilangan prima dan kedua bilangan tersebut tidak boleh sama besar;

b. *User* menghitung nilai N;

$$\boldsymbol{N = P.Q} \qquad (1)$$

Di mana :
P = bilangan prima acak tidak sama dengan Q.
Q = bilangan prima acak tidak sama dengan P.
N = kunci *public* untuk enkripsi dan dekripsi.

c. Setelah mendapatkan nilai N, selanjutnya *user* akan menentukan nilai $\phi N$;

$$\phi N = (P - 1).(Q - 1) \quad (2)$$

Di mana :

$\phi N$ = nilai untuk mencari kunci enkripsi dan dekripsi.

d. Proses selanjutnya yaitu mencari nilai D yang merupakan kunci *public*;

$$D = \frac{1 + i\phi(N)}{E} \quad (3)$$

Di mana :

D = kunci dekripsi

i = 1, 2, 3, ... N

E = bilangan prima acak < $\phi N$, digunakan untuk kunci enkripsi

e. Selanjutnya *user* akan menentukan nilai E yaitu nilai kunci *private* yang bernilai prima dan memiliki ketentuan $1 < E < \phi N$;

$$(E.D)m \quad \phi(N) = 1 \quad (4)$$

f. Tentukan nilai Initialization Vector;

$$I = P + Q \quad (5)$$

g. Kunci enkripsi = {E, N} dan kunci dekripsi = {D, N}.

Setelah kunci dibangkitkan, selanjutnya adalah proses enkripsi. Proses enkripsi RSA menggunakan Persamaan 6 dan proses dekripsi.

$$C = P^E m \quad N \quad (6)$$

$$P = C^D m \quad N \quad (7)$$

Di mana :

C = kode desimal *cipher text*

P = kode desimal *plain text*

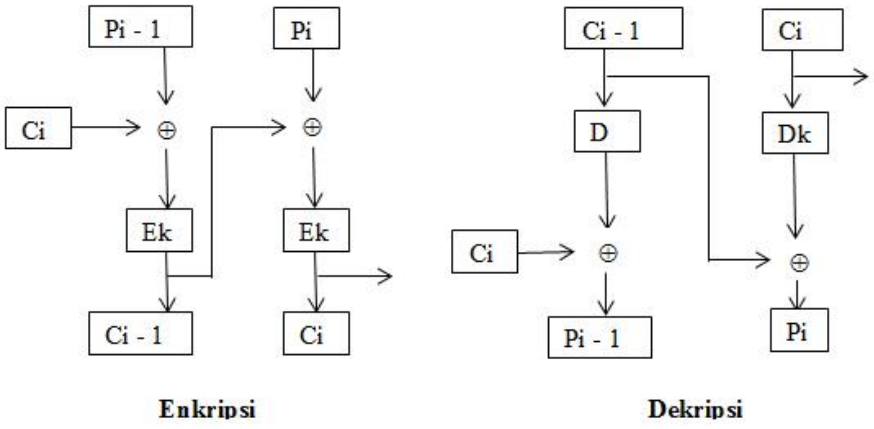
E = kunci enkripsi

D = kunci dekripsi

N = kunci *public*

### 1.2.5 Cipher Block Chaining

Metode kriptografi ini menerapkan mekanisme penggunaan kembali pada sebuah blok, yang dalam hal ini hasil enkripsi blok sebelumnya digunakan kembali dalam enkripsi blok setelahnya. Hasil enkripsi dari blok sebelumnya akan berpegaruh dan digunakan pada proses enkripsi selanjutnya. Setiap blok *cipher text* bukan hanya bergantung pada blok *plain text*, tetapi bergantung pula pada blok-blok *plain text* sebelumnya sehingga *plain text* yang sama belum tentu menghasilkan *cipher text* yang sama pula. Skema pengamanan dari CBC dapat dilihat pada Gambar 4.




**Gambar 2. Skema enkripsi dan dekripsi CBC**

Pada Gambar 4, untuk menghasilkan blok *cipher* pertama, Ci digunakan untuk menggantikan blok *cipher text* sebelumnya. Sebaliknya pada dekripsi, blok *plain text* pertama diperoleh dengan cara melakukan operasi XOR antara Ci dengan hasil dekripsi terhadap blok *cipher text* pertama. Proses enkripsi dan dekripsi metode CBC dinyatakan dalam Persamaan (8) dan Persamaan (9).

$$C = E\ (P\ \text{ă}\ C - 1) \quad (8)$$

$$P = D\ (C\ \text{ă}\ C - 1) \quad (9)$$

Pada proses enkripsi dan dekripsi CBC ada nilai yang disebut Initialization Vector (IV) yang digunakan untuk membuat *feedback* dari setiap karakter, untuk mencari nilai IV ini adalah secara acak.

Tahapan proses enkripsi CBC :

a. Bagi *plain text* menjadi blok yang telah ditentukan;
b. Tiap blok yang telah dibagi kemudian dilakukan operasi XOR dengan Ci;
c. Hasil yang didapat kemudian dilakukan operasi XOR kembali dengan kunci;
d. Hasil operasi XOR tersebut digeser 1 *bit* ke kiri;
e. Hasil tersebut menjadi Ci untuk blok berikutnya;

Tahapan proses dekripsi CBC:

a. Dilakukan dari blok terakhir;
b. Geser 1 *bit* ke kanan;
c. XOR dengan kunci;

XOR dengan blok *cipher text* sebelumnya.

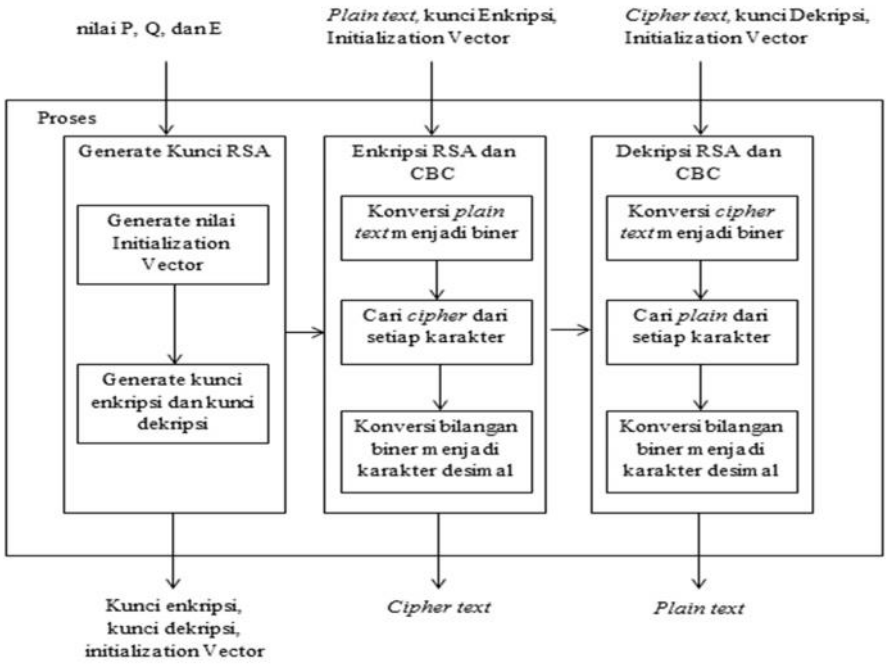
## 2. PEMBAHASAN

### 2.1 Perancangan Sistem

Sistem ini dibangun dengan menggabungkan dua metode kriptografi untuk *database* MySQL, Pada Gambar 5 diperlihatkan bahwa proses pertama yang dilakukan adalah melakukan pembuatan kunci enkripsi dan dekripsi, kunci ini didapat dari proses perhitungan dengan menggunakan algoritma RSA dengan masukan berupa tiga angka prima yang bernilai beda, keluaran dari proses ini adalah kunci enkripsi yang digunakan untuk melakukan proses enkripsi, kunci dekripsi yang digunakan dalam proses dekripsi dan kunci IV yang digunakan dalam proses enkripsi maupun dekripsi.

Proses selanjutnya yaitu proses enkripsi, proses ini menggunakan dua metode yaitu RSA terlebih dahulu, kemudian hasil enkripsi dari RSA akan diproses kembali dengan metode CBC. Masukan dari proses enkripsi yaitu kunci enkripsi, kunci IV, dan *plain text* yang didapat dari data tabel yang akan dienkripsi pada *database.* Hasil keluaran dari proses ini adalah berupa *cipher text* yaitu kumpulan karakter acak yang tidak dapat dipahami.

Proses terakhir yaitu proses dekripsi, proses ini menggunakan metode CBC terlebih dahulu, kemudian setelah proses CBC selesai akan dilakukan proses dekripsi dengan metode RSA untuk mengetahui isi dari data yang telah dienkrip.




**Gambar 5. Gambaran sistem pengamanan data user**

### 2.2 Implementasi Perhitungan Kunci Enkripsi dan Dekripsi

Dalam menentukan kunci enkripsi dan dekripsi, sistem menggunakan algoritma RSA untuk menentukannya. Algoritma RSA juga akan digunakan pada saat proses enkripsi dan dekripsi, algoritma ini menggunakan kunci yang berbeda pada saat enkripsi dan dekripsi, hal tersebut membuat pengamanan akan lebih ketat karena kerumitan dalam penentuan kuncinya.

Untuk menentukan kunci *public* RSA dibutuhkan beberapa parameter masukan yaitu nilai P, Q, dan E. Nilai P, Q, dan E boleh berbeda, nilai P dan Q adalah sebuah bilangan prima yang tidak boleh sama, sedangkan nilai E adalah nilai yang merupakan bilangan prima. Untuk menentukan kunci enkripsi dan dekripsi, yang pertama cari dulu nilai dari N dengan Persamaan (1).

$$N = 13 . 17 = 221$$

Nilai N akan digunakan untuk kunci *public*, kunci *public* ini nantinya akan terlibat dalam perhitungan enkripsi maupun perhitungan dekripsi. Selanjutnya menentukan nilai $\phi N$ dengan menggunakan Persamaan (2).

$$\phi N = (13 - 1).(17 - 1) = 192$$

Nilai $\phi N$ digunakan untuk mencari kunci enkripsi dan dekripsi, nilai $\phi N$ ini juga digunakan dalam mencari kesesuaian antara kunci enkripsi dan dekripsi. Selanjutnya menentukan kunci dekripsi dengan Persamaan (3).

$$D = \frac{1 + i\phi(192)}{7} = 55$$

Kunci dekripsi digunakan untuk menerjemahkan *cipher text* menjadi *plain text*, nilai ini selanjutnya akan dilakukan penyesuaian terhadap kunci enkripsi apakah dapat digunakan atau tidak. Nilai kunci enkripsi harus relatif prima dengan $\varphi$ , cara mengetahuinya adalah dengan mengunakan Persamaan (4). Jika hasilnya ekivalen maka nilai E dapat digunakan.

$$(7 . 55) m \quad 92 = 1$$

Tahap selanjutnya adalah menentukan nilai Initialization Vector (IV), nilai IV ini digunakan dalam algoritma CBC untuk mencari *feedback* dari setiap karakter yang dienkrip. Biasanya untuk mencari nilai ini adalah acak. Dalam penelitian ini nilai IV didapatkan dari Persamaan (5).

$$I = 13 + 17 = 30$$

Apabila proses perhitungan berhasil maka kunci enkripsi dan dekripsinya adalah {7, 221} dan {55, 221} serta IV adalah {30}. Selanjutnya nilai-nilai yang sudah didapatkan akan digunakan dalam proses enkripsi dan dekripsi.

### 2.3 Perhitungan Enkripsi dan Dekripsi

Untuk melakukan enkripsi dan dekripsi, sistem ini menggunakan algoritma RSA dan CBC. Hal pertama yang diakukan dalam proses enkripsi adalah mencari nilai desimal dari masing-masing karakter, selanjutnya semua karakter tersebut akan diubah ke dalam *cipher text* dengan algoritma RSA. Setelah itu, algoritma CBC akan digunakan untuk mencari nilai IV dari setiap karakter sehingga saat proses enkripsi setiap nilai IV yang digunakan akan berbeda. Untuk proses dekripsi, langkah-langkah yang digunakan tidak berbeda jauh dengan proses enkripsi, yang membedakan hanya kunci yang digunakan dalam proses perhitungannya.

Misalkan untuk melakukan enkripsi pada kata “Unjani” maka setiap karakter akan diubah terlebih dulu ke dalam bilangan desimal, selanjutnya akan digunakan Persamaan (6) untuk enkripsi dan Persamaan (7) untuk dekripsi. Bilangan desimal yang digunakan adalah yang berstandar American Standart Code for Infomation Interchange (ASCII), kode ASCII merupakan kode yang digunakan untuk memfasilitasi interaksi antara pengguna dengan komputer. Dengan kata lain, ASCII digunakan untuk pertukaran data dan komunikasi data dengan mengubah angka menjadi karakter. Untuk mengetahui kode ASCII dari masing-masing karakter, digunakan tabel ASCII untuk menerjemahkan karakter menjadi kde ASCII.

#### 2.3.1 Enkripsi RSA dan Cipher Block Chaining

Untuk melakukan enkripsi, *plain text* terlebih dahulu akan dikonversi ke dalam kode ASCII dalam bentuk desimal. Data hasil konversi dapat dilihat pada Tabel 1.

**Tabel 1. Konversi plain text ke ASCII**

| *Karakter* | *Kode ASCII* |
|---|---|
| U | 85 |
| n | 110 |
| j | 106 |
| a | 97 |
| n | 110 |
| i | 105 |

Langkah selanjutnya adalah menghitung setiap karakter untuk mengubahnya menjadi karakter *cipher* dengan menggunakan Persamaan (8), setiap karakter akan dipangkatkan dengan kunci enkripsi kemudian dilakukan

proses untuk mencari hasil bagi terhadap nilai n, hasil operasi tersebut kemudian akan dilakukan proses XOR dengan nilai IV, hasil dari opersi tersebut adalah nilai desimal dari karakter *cipher* sekaligus nilai IV untuk proses enkripsi berikutnya, proses tersebut diulangi sampai karakter habis. Hasil konversi bilangan decimal menjadi karakter *cipher* dapat dilihat pada Tabel 2.

$C0 = (85^7 \; m \quad 221) \; \square \; 30 = 150$
$C1 = (110^7 \; m \quad 221) \; \square \; 150 = 92$
$C2 = (106^7 \; m \quad 221) \; \square \; 92 = 47$
$C3 = (97^7 \; m \quad 221) \; \square \; 47 = 40$
$C4 = (110^7 \; m \quad 221) \; \square \; 40 = 226$
$C5 = (73^7 \; m \quad 221) \; \square \; 226 = 173$

**Tabel 2. Konversi kode ASCII ke cipher text**

| *Kode ASCII* | *Karakter Cipher* |
|---|---|
| 150 | – |
| 92 | \ |
| 47 | / |
| 40 | ( |
| 226 | â |
| 173 | - |

### 2.3.2 Dekripsi RSA dan Cipher Block Chaining

Karakter cipher yang akan didekripsi akan dikonversi kembali menjadi kode ASCII yang bertipe desimal, proses dekripsi dimulai dari karakter terakhir. Konversi karakter *cipher* menjadi bilangan decimal dapat dilihat pada Tabel 3.

**Tabel 3. Konversi cipher ke kode ASCII**

| *Cipher text* | *Kode ASCII* |
|---|---|
| – | 150 |
| \ | 92 |
| / | 47 |
| ( | 40 |
| Â | 226 |
| - | 173 |

Langkah selanjutnya adalah menghitung setiap karakter untuk mengubahnya menjadi karakter *plain* dengan menggunakan Persamaan (9), setiap karakter akan dilakukan operasi XOR dengan nilai IV kemudian dipangkatkan dengan kunci dekripsi, hasil operasi tersebut kemudian akan dilakukan proses modulus dengan nilai n, hasil dari opersi tersebut adalah nilai desimal dari karakter *plain* sekaligus nilai IV untuk proses dekripsi berikutnya, proses tersebut diulangi sampai karakter habis. Hasil konversi bilangan decimal menjadi karakter *plain* dapat dilihat pada Tabel 4.

$P5 = (173 \; \square \; 226)^5 \mod 221 = 105$
$P4 = (226 \; \square \; 40)^5 \mod 221 = 110$
$P3 = (40 \; \square \; 47)^5 \mod 221 = 97$
$P2 = (47 \; \square \; 92)^5 \mod 221 = 106$
$P1 = (92 \; \square \; 150)^5 \mod 221 = 110$
$P0 = (150 \; \square \; 30)^5 \mod 221 = 85$

**Tabel 4. Konversi kode ASCII ke plain text**

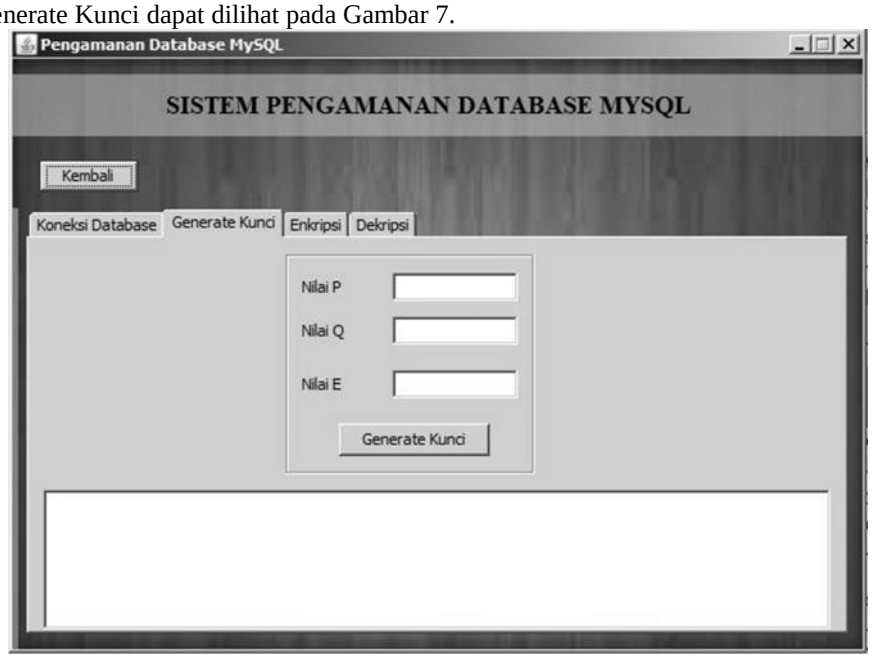
| *Kode ASCII* | *Karakter* |
|---|---|
| 85 | U |
| 110 | n |
| 106 | j |
| 97 | a |
| 110 | n |
| 105 | i |

### 2.4 Implementasi Sistem

Penelitian ini diimplementasikan dalam sebuah perangkat lunak berbasis *desktiop*. Fitur koneksi *database* ini memungkinkan *user* untuk melakukan koneksi dengan *database* yang akan diamankan, kemudian memilih tabel yang tersedia pada *database* yang telah dipilih sebelumnya. Fitur koneksi *database* dapat dilihat pada Gambar 6.


**Gambar 6. Koneksi database**

Setelah melakukan koneksi, selanjutnya *user* dapat menentukan kunci yang digunakan untuk enkripsi dan dekripsi. Fitur Generate Kunci dapat dilihat pada Gambar 7.



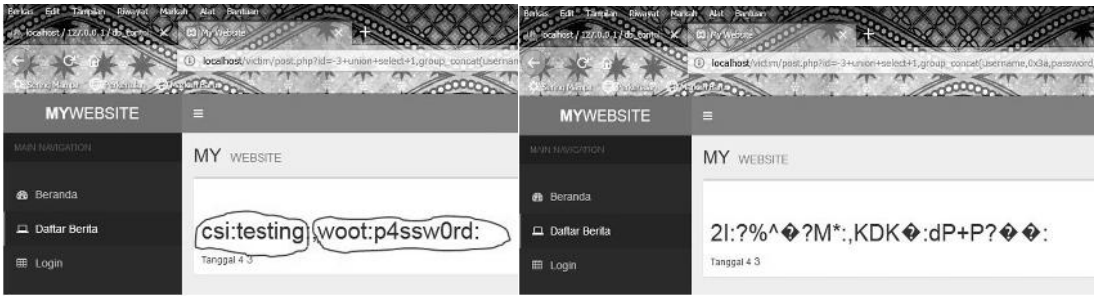

**Gambar 7. Fitur generate kunci**

Fitur selanjutnya adalah fitur enkripsi, *user* dapat melakukan pengamanan terhadap tabel pada fitur ini. Tampilan fitur enkripsi data dapat dilihat pada Gambar 8.


**Gambar 8. Fitur enkripsi data**

## 2.5 Hasil Penelitian

Setelah dilakukan pengamanan terhadap tabel pada *database* menggunakan sistem ini, data yang dipilih berhasil dienkripsi. Pengujian dilakukan untuk melihat hasil dari proses pengamanan, pengujian yang dilakukan adalah melakukan pengamanan terlebih dahulu terhadap tabel *user* pada salah satu sistem, kemudian sistem tersebut diretas untuk melihat hasil pengamanan yang dilakukan. Hasil pengamanan tersebut dapat dilihat pada Gambar 9.




**Gambar 9. Hasil proses SQL Injection sebelum proses enkripsi (kiri) dan setelah proses dekripsi (kanan)**

## 3. KESIMPULAN

Hasil dari penelitian ini adalah sistem yang dapat mengamankan data pada sebuah *database* yang salah satu tabelnya adalah tabel *user*. Setelah dilakukan proses enkripsi pada tabel, seluruh data pada tabel berhasil dienkripsi sehingga data di dalam tabel menjadi tidak beraturan dan tidak dapat dibaca, pada saat *database* diretas dengan SQL Injection pun data yang tampil adalah data yang telah terenkripsi, data yang terenkripsi kemudian didekripsi dan hasilnya data berhasil kembali seperti sebelum dienkripsi.

## PUSTAKA

Agrawal, H., & Sharma, M. 2014. A review of text encryption techniques. *Asian Journal of Computer Science and Information Technology*, *4*(5), 47–54.

Arya, P. K., Aswal, M. S., & Kumar, V. 2012. Comparative Study of Asymmetric Key Cryptographic Algorithms. *International Journal of Computer Science & Communication Networks*, *5*(1), 17–21.

Basri. 2016. Kriptografi Simetris Dan Asimetris Dalam Perspektif Keamanan Data Dan Kompleksitas Komputasi. *Jurnal Ilmiah Ilmu Komputer*, *2*(2), 17–23.

Chandra. 2016. Keamanan Data Dengan Metode Kriptografi Kunci Publik. *Jurnal Times*, *2*(2), 11–15.

Galla, L. K., Koganti, V. S., & Nuthalapati, N. (2017). Implementation of RSA. In *International Conference on Control Instrumentation Communication and Computational Technologies* (pp. 81–87).

Gupta, S., & Sharma, J. 2012. A Hybrid Encryption Algorithm based on RSA and Diffie-Hellman. In *International Conference on Computational Intelligence and Computing Research* (Vol. 13).

Henry, Kridalaksana, A. H., & Arifin, Z. 2016. Kriptografi Aes Mode Cbc Pada Citra Digital Berbasis Android. In *Seminar Ilmu Komputer dan Teknologi Informasi* (Vol. 1, pp. 45–52).

Lal, N. A. 2017. A Review Of Encryption Algorithms-RSA And Diffie-Hellman. *International Journal of Scientific & Technology Research*, *6*(07), 84–87.

Lestiawan, H., & Purnama, R. D. O. 2016. Pengamanan Dokumen Teks Menggunakan Algoritma Kriptografi Mode Operasi Cipher Block Chaining (Cbc) Dan Steganografi Metode End of File (Eof). *Techno.COM*, *15*(1), 22–31.

Lombu, D., Tarihoran, S. D., & Gulo, I. 2018. Kombinasi Mode Cipher Block Chaining Dengan Algoritma Triangle Chain Cipher Pada Penyandian Login Website. *Jurnal Sains Komputer Dan Informatika*, *2*(1), 1–11.

Menon, C. B., Joy, A., Emmanuel, E., & Paul, V. 2017. Analysis on Symmetric Algorithms. *International Journal of Engineering Science and Computing*, *7*(3), 5285–5289.

Rochmah, N., & Ardiansyah. 2013. Desain Kriptografi CBC Modifikasi pada Proses Pengamanan Pesan melalui Email. In *Seminar Nasional Teknologi Informasi dan Multimedia* (pp. 1–6).

Shende, V., Sudi, G., & Kulkarni, M. 2017. Fast Cryptanalysis of RSA Encrypted Data using A Combination of Mathematical and Brute Force Attack in Distributed Computing Environment. In *International Confernce on Power, Control, Signals and Instrumentation Engineering* (pp. 2446–2449).

Tripathi, R., & Sanjay, A. 2014. Comparative Study of Symmetric and Asymmetric Cryptography Techniques Ritu. *International Journal of Advance Foundation and Research in Computer*, *1*(6), 2348–4853.

Wulansari, D., Muslim, M. A., & Sugiharti, E. 2016. Implementation of RSA Algorithm with Chinese Remainder Theorem for Modulus N 1024 Bit and 4096 Bit. *International Journal of Computer Science and Security*, *10*(5), 186–194.

Yassein, M. B., Aljawarneh, S., Qawasmeh, E., Mardini, W., & Khamayseh, Y. (2017). Comprehensive study of symmetric key and asymmetric key encryption algorithms. In *International Conference on Engineering and Technology* (pp. 1–7).