



**MENINGKATKAN KEAMANAN DATA MENGGUNAKAN OPERASI  
XOR MODE ALGORITMA CIPHER BLOCK CHAINING (CBC)  
DENGAN KUNCI ACAK BLUM BLUM SHUB**

Disusun dan Diajukan untuk Memenuhi Persyaratan Ujian Akhir Memperoleh  
Gelar Sarjana Komputer pada Fakultas Sains dan Teknologi  
Universitas Pembangunan Panca Budi  
Medan

---

**SKRIPSI**

---

**OLEH**

**NAMA : WINDI ASMITA**  
**NPM : 1514370082**  
**PROGRAM STUDI : SISTEM KOMPUTER**

**FAKULTAS SAINS DAN TEKNOLOGI  
UNIVERSITAS PEMBANGUNAN PANCA BUDI  
MEDAN  
2019**

**LEMBAR PENGESAHAN**

**MENINGKATKAN KEAMANAN DATA MENGGUNAKAN OPERASI  
XOR MODE ALGORITMA CIPHER BLOCK CHAINING (CBC)  
DENGAN KUNCI ACAK BLUM BLUM SHUB**

**Disusun Oleh:**


**NAMA : WINDI ASMITA  
NPM : 1514370082  
PROGRAM STUDI : SISTEM KOMPUTER**

**Skripsi Telah Disetujui oleh Dosen Pembimbing Skripsi  
Pada Tanggal : 16 Desember 2019**

**Dosen Pembimbing I**

**Dosen Pembimbing II**

  
**Eko Hariyanto, S.Kom., M.Kom**

  
**Supina Batubara, S.Kom., M.Kom**

**Mengetahui,**

**Dekan Fakultas Sains Dan Teknologi**

**Ketua Program Studi Sistem Komputer**



  
**Eko Hariyanto, S.Kom., M.Kom**

## SURAT PERNYATAAN

Saya yang bertanda tangan di bawah ini :

Nama : Windi Asmita  
NPM : 1514370082  
Prodi : Sistem Komputer  
Konsentrasi : Keamanan Jaringan Komputer  
Judul Skripsi : Meningkatkan Keamanan Data Menggunakan Operasi XOR Mode Algoritma Cipher Block Chaining (CBC) dengan Kunci Acak Blum Blum Shub.

Dengan ini menyatakan bahwa :

1. Tugas Akhir/Skripsi saya bukan hasil Plagiat.
2. Saya tidak akan menuntut perbaikan nilai Indeks Prestasi Kumulatif (IPK) setelah ujian Sidang Meja Hijau.
3. Skripsi saya dapat dipublikasikan oleh pihak lembaga dan saya tidak akan menuntut akibat publikasi tersebut

Demikian pernyataan ini saya perbuat dengan sebenar-benarnya, terima kasih.

Medan, 16 Desember 2019

Yang membuat pernyataan



**Windi Asmita**  
**1514370082**

## PERNYATAAN ORISINALITAS

Dengan ini menyatakan bahwa dalam skripsi ini tidak terdapat karya yang diajukan untuk memperoleh gelar kesarjanaan di dalam perguruan tinggi, dan sepanjang pengetahuan saya juga tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis didalam skripsi ini dan disebutkan dalam daftar pustaka.

Medan, 16 Desember 2019

Yang membuat pernyataan

METERAI  
TEMPEL  
FB83EAEF81597398  
6000  
ENAM RIBURUPIAH



Winda Asmita  
1514370082



Telah Diperiksa oleh LPMU  
dengan Plagiarisme 10 %

Medan, 07 NOV 2019

FM-BPAA-2012-041

Hal : Permohonan Meja Hijau



Medan, 07 November 2019  
Kepada Yth : Bapak/Ibu Dekan  
Fakultas SAINS & TEKNOLOGI  
UNPAB Medan  
Di -  
Tempat

Telah di terima  
berkas persyaratan  
dapat di proses  
Medan 12 / 11 / 2019

BPA  
TEGUH WAHYONO, SE., MM.

Dengan hormat, saya yang bertanda tangan di bawah ini :

Nama : Windi Asmita  
Tempat/Tgl. Lahir : Minta Kasih / 31 Maret 1998  
Nama Orang Tua : SUWANDI  
N. P. M : 1514370082  
Fakultas : SAINS & TEKNOLOGI  
Program Studi : Sistem Komputer  
No. HP : 085359649513  
Alamat : Jl. Binjai Bahorok Kab. Langkat

Datang bermohon kepada Bapak/Ibu untuk dapat diterima mengikuti Ujian Meja Hijau dengan judul MENINGKATKAN KEAMANAN DATA MENGGUNAKAN OPERASI XOR MODE ALGORITMA CIPHER BLOK CHAINING (CBC) DENGAN KUNCI ACAK BLUM-BLUM SHUB, Selanjutnya saya menyatakan :

1. Melampirkan KKM yang telah disahkan oleh Ka. Prodi dan Dekan
2. Tidak akan menuntut ujian perbaikan nilai mata kuliah untuk perbaikan indek prestasi (IP), dan mohon diterbitkan ijazahnya setelah lulus ujian meja hijau.
3. Telah tercap keterangan bebas pustaka
4. Terlampir surat keterangan bebas laboratorium
5. Terlampir pas photo untuk ijazah ukuran 4x6 = 5 lembar dan 3x4 = 5 lembar Hitam Putih
6. Terlampir foto copy STTB SLTA dilegalisir 1 (satu) lembar dan bagi mahasiswa yang lanjutan D3 ke S1 lampirkan ijazah dan transkripnya sebanyak 1 lembar.
7. Terlampir pelunasan kwintasi pembayaran uang kuliah berjalan dan wisuda sebanyak 1 lembar
8. Skripsi sudah dijilid lux 2 exemplar (1 untuk perpustakaan, 1 untuk mahasiswa) dan jilid kertas jeruk 5 exemplar untuk penguji (bentuk dan warna penjilidan diserahkan berdasarkan ketentuan fakultas yang berlaku) dan lembar persetujuan sudah di tandatangani dosen pembimbing, prodi dan dekan
9. Soft Copy Skripsi disimpan di CD sebanyak 2 disc (Sesuai dengan Judul Skripsinya)
10. Terlampir surat keterangan BKKOL (pada saat pengambilan ijazah)
11. Setelah menyelesaikan persyaratan point-point diatas berkas di masukan kedalam MAP
12. Bersedia melunaskan biaya-biaya uang dibebankan untuk memproses pelaksanaan ujian dimaksud, dengan perincian sbb :

1. [102] Ujian Meja Hijau	: Rp.	100.000
2. [170] Administrasi Wisuda	: Rp.	1,500,000
3. [202] Bebas Pustaka	: Rp.	100,000
4. [221] Bebas LAB	: Rp.	5,000
<b>Total Biaya</b>	<b>: Rp.</b>	<b>1.705.000</b>

12/11/19  
W

Periode Wisuda Ke : 64

Ukuran Toga : M



Hormat Saya

Windi Asmita  
1514370082

Catatan :

- 1. Surat permohonan ini sah dan berlaku bila ;
  - o a. Telah dicap Bukti Pelunasan dari UPT Perpustakaan UNPAB Medan.
  - o b. Melampirkan Bukti Pembayaran Uang Kuliah aktif semester berjalan
- 2. Dibuat Rangkap 3 (tiga), untuk - Fakultas - untuk BPAA (asli) - Mhs.ybs.



# Plagiarism Detector v. 1281 - Originality Report

## "WINDI ASMITA\_1514370082\_SISTEM KOMPUTER.docx"

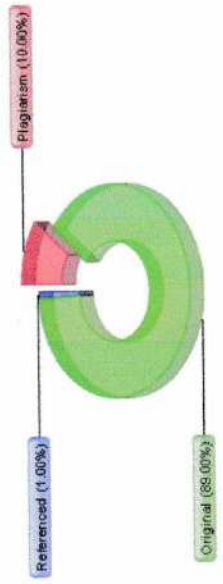
Analyzed document: 04/11/2019 16:53:01

Check Type: Internet - via Google and Bing

Licensed to: Universitas Pembangunan Panca Budi\_License2



Relation chart:



Distribution graph:



Comparison Preset: Rewrite. Detected language: Indonesian

Top sources of plagiarism:

Source	Percentage	Words
<a href="http://jurnek.akprind.ac.id/sites/default/files/04-08_rahman.pdf">http://jurnek.akprind.ac.id/sites/default/files/04-08_rahman.pdf</a>	% 4	wrds: 702
<a href="https://indobahan.blogspot.com/2011/03/spa-sih-kriptograf-simetris-asimetris.ht...">https://indobahan.blogspot.com/2011/03/spa-sih-kriptograf-simetris-asimetris.ht...</a>	% 2	wrds: 314
<a href="https://www.duniainkom.com/pengertian-dan-jungai-php-dalam-pemrograman-web">https://www.duniainkom.com/pengertian-dan-jungai-php-dalam-pemrograman-web</a>	% 2	wrds: 242





# UNIVERSITAS PEMBANGUNAN PANCA BUDI FAKULTAS SAINS & TEKNOLOGI

Jl. Jend. Gatot Subroto Km 4,5 Medan Fax. 061-8458077 PO.BOX : 1099 MEDAN

PROGRAM STUDI TEKNIK ELEKTRO	(TERAKREDITASI)
PROGRAM STUDI ARSITEKTUR	(TERAKREDITASI)
PROGRAM STUDI SISTEM KOMPUTER	(TERAKREDITASI)
PROGRAM STUDI TEKNIK KOMPUTER	(TERAKREDITASI)
PROGRAM STUDI AGROTEKNOLOGI	(TERAKREDITASI)
PROGRAM STUDI PETERNAKAN	(TERAKREDITASI)

## PERMOHONAN JUDUL TESIS / SKRIPSI / TUGAS AKHIR\*

Yang bertanda tangan di bawah ini :

Nama Lengkap : Windi Asmita  
 Tempat/Tgl. Lahir : mintakasih / 31 Maret 1998  
 Nomor Pokok Mahasiswa : 1514370082  
 Program Studi : Sistem Komputer  
 Konsentrasi : Keamanan Jaringan Komputer  
 Jumlah Kredit yang telah dicapai : 141 SKS, IPK 3.35  
 Nomor Hp : 085359649513  
 Saya ini mengajukan judul sesuai bidang ilmu sebagai berikut :

### Judul

MENINGKATKAN KEAMANAN DATA DENGAN MENGGUNAKAN OPERASI XOR MODE ALGORITMA CIPHER BLOK CHAINING (CBC) DENGAN KUNCI ACAK BLUM-BLUM SHUB


Diisi Oleh Dosen Jika Ada Perubahan Judul

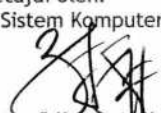
Yang Tidak Perlu


  
 (Ir. Bhakti Alamsyah, M.T., Ph.D.)

20 Februari  
 Medan, 20 Oktober 2019  
 Remohon,  
  
 (Windi Asmita)

Tanggal : .....  
 Disahkan oleh  
 Dekan  
  
 (Sri Shindi Indira, S.T., M.Sc.)

Tanggal : 19/10/2019 .....  
 Disetujui oleh :  
 Dosen Pembimbing I :  
  
 ( Eko Hariyanto, S.Kom., M.Kom )

Tanggal : 19/02/2019 .....  
 Disetujui oleh:  
 Ka. Prodi Sistem Komputer  
  
 ( Eko Hariyanto, S.Kom., M.Kom )

Tanggal : 19/02/2019 .....  
 Disetujui oleh:  
 Dosen Pembimbing II:  
  
 ( Supina Batubara, S.Kom., M.Kom )



UNIVERSITAS PEMBANGUNAN PANCA BUDI  
**FAKULTAS SAINS & TEKNOLOGI**  
 Jl. Jend. Gatot Subroto Km. 4,5 Telp (061) 8455571  
 website : www.pancabudi.ac.id email: unpab@pancabudi.ac.id  
 Medan - Indonesia

Universitas : Universitas Pembangunan Panca Budi  
 Fakultas : SAINS & TEKNOLOGI  
 Pembimbing I : EKO HARYANTO, S.Kom, M.Kom  
 Pembimbing II : SUPINA BATUBARA, S.Kom, M.Kom  
 Mahasiswa : WINDI ASMITA  
 Jurusan/Program Studi : Sistem Komputer  
 NPM/Pokok Mahasiswa : 1514370082  
 Bidang Pendidikan : SI  
 Tugas Akhir/Skripsi : Meningkatkan Keamanan Data Menggunakan Operasi XOR Mode Algoritma Cipher Block Chaining (CBC) Dengan Kunci Acak Blum Blum Shub

TANGGAL	PEBAHASAN MATERI	PARAF	KETERANGAN
02/2019	Acc Sempro	<i>[Signature]</i>	
04/19	* sesuaikan rumusan masalah dengan latar belakang	<i>[Signature]</i>	BAB I
05/19	* Acc BAB I	<i>[Signature]</i>	
06-19	* tambahkan fitur deskripsi pada program	<i>[Signature]</i>	
07-19	* perbaiki tampilan program	<i>[Signature]</i>	
08-19	* Acc program	<i>[Signature]</i>	
09-19	* Acc Seminar hasil	<i>[Signature]</i>	

Medan, 17 Oktober 2019

Diketahui/Ditetujui oleh :

Dekan



Sri Shindi Indira, S.T., M.Sc.





UNIVERSITAS PEMBANGUNAN PANCA BUDI  
**FAKULTAS SAINS & TEKNOLOGI**

Jl. Jend. Gatot Subroto Km. 4,5 Telp (061) 8455571  
 website : www.pancabudi.ac.id email: unpab@pancabudi.ac.id  
 Medan - Indonesia

Universitas : Universitas Pembangunan Panca Budi  
 Fakultas : SAINS & TEKNOLOGI  
 Pembimbing I : EKO HARIYANTO S.kom, M.kom  
 Pembimbing II : SUPINA BATUBARA S.kom, M.kom  
 Mahasiswa : WINDI ASMITA  
 Program Studi : Sistem Komputer  
 Pokok Mahasiswa : 1514370082  
 Pendidikan : ST  
 Tugas Akhir/Skripsi : Meningkatkan keamanan data menggunakan operasi XOR Mode Algoritma Cipher Block Chaining (CB) Dengan Kunci Acak Blum-Blum Shub

WAKTU	PEMBAHASAN MATERI	PARAF	KETERANGAN
10-19	Ace sedang mega lujan	<i>[Signature]</i>	
11-19	Ace jibid Guripis	<i>[Signature]</i>	

Medan, 17 Oktober 2019

Diketahui/Disetujui oleh :



Sri Shindi Indira, S.T., M.Sc.



**UNIVERSITAS PEMBANGUNAN PANCA BUDI**  
**FAKULTAS SAINS & TEKNOLOGI**  
 Jl. Jend. Gatot Subroto Km. 4,5 Telp (061) 8455571  
 website : www.pancabudi.ac.id email: unpab@pancabudi.ac.id  
 Medan - Indonesia

Universitas : Universitas Pembangunan Panca Budi  
 Fakultas : SAINS & TEKNOLOGI  
 Dosen Pembimbing I : Eko Hariyanto, S.KOM, M.KOM  
 Dosen Pembimbing II : Supriana Patubera, S.KOM, M.KOM  
 Nama Mahasiswa : WINDI ASMITA  
 Jurusan/Program Studi : Sistem Komputer  
 Nomor Pokok Mahasiswa : 1514370082  
 Bidang Pendidikan : SI  
 Judul Tugas Akhir/Skripsi : Meningkatkan Keamanan Data Dengan menggunakan Algoritma XOR Pada mode Operasi CBC (Cipher Block Chaining) Dengan Kunci acak Blum-Blum-Shub

TANGGAL	PEMBAHASAN MATERI	PARAF	KETERANGAN
02/2019	Acc Sempr	<i>[Signature]</i>	
05/2019	Revisi Bab I, (bagian Bab II)	<i>[Signature]</i>	
10/05/2019	Revisi Bab II	<i>[Signature]</i>	
07/2019	Revisi Bab II (Dapat setelah nilai kredit)	<i>[Signature]</i>	
08/2019	Acc Bab II	<i>[Signature]</i>	
08/2019	Revisi Bab II (melalui kelompok dan)	<i>[Signature]</i>	
03/2019	Acc Bab II (bagian Bab II)	<i>[Signature]</i>	
10/2019	Acc Sempr	<i>[Signature]</i>	
10/2019	Acc Endang Stage Abgan	<i>[Signature]</i>	
12/2019	Menyempurnakan lampiran	<i>[Signature]</i>	
1/2019	Acc final	<i>[Signature]</i>	

Medan, 21 Februari 2019

Diketahui/Ditetujui oleh :

Dekan



Sri Shinda Indira, S.T., M.Sc.



YAYASAN PROF. DR. H. KADIRUN YAHYA  
UNIVERSITAS PEMBANGUNAN PANCA BUDI  
**LABORATORIUM KOMPUTER**  
Jl. Jend. Gatot Subroto Km 4,5 Sei Sikambing Telp. 061-8455571  
Medan - 20122

**KARTU BEBAS PRAKTIKUM**

Yang bertanda tangan dibawah ini Ka. Laboratorium Komputer dengan ini menerangkan bahwa :

Nama : Windi Asmita  
N.P.M. : 1514370082  
Tingkat/Semester : Akhir  
Fakultas : SAINS & TEKNOLOGI  
Jurusan/Prodi : Sistem Komputer

Benar dan telah menyelesaikan urusan administrasi di Laboratorium Komputer Universitas Pembangunan Panca Budi Medan.

Medan, 06 November 2019

20/ Ka. Laboratorium





## ABSTRAK

WINDI ASMITA

**Meningkatkan Keamanan Data Menggunakan Operasi XOR Mode  
Algoritma Cipher Block Chaining (CBC) dengan Kunci Acak Blum Blum  
Shub  
2019**

Penelitian ini bertujuan untuk mengkombinasikan teknik XOR dengan mode operasi *Cipher Block Chaining* (CBC) 256-bits dengan kunci yang dibangkitkan secara acak sepanjang 256-bits dengan algoritma Blum Blum Shub. Kombinasi algoritma yang dihasilkan diimplementasikan dengan bahasa pemrograman PHP untuk kemudian diuji dan dianalisis hasil proses enkripsi dan dekripsi terhadap sampel pesan yang digunakan. Hasil penelitian menunjukkan bahwa waktu proses yang dibutuhkan baik untuk enkripsi dan dekripsi sangat cepat, dan perangkat yang dibutuhkan untuk menjalankan aplikasi juga sangat ringan, sehingga algoritma yang dirancang dikategorikan hemat dalam hal sumber daya komputer dan waktu proses yang dibutuhkan. Hasil analisis keamanan juga menunjukkan bahwa dibutuhkan waktu  $3,6717 \times 10^{49}$  Juta Tahun untuk memecahkan kunci dengan menggunakan teknik brute force dan asumsi  $10^{14}$  percobaan perdetik. Cipher text yang dihasilkan juga sangat sensitif terhadap perubahan walau hanya 1 bit data yang dapat merusak seluruh *cipher text* setelahnya, sehingga keaslian *cipher text* juga terjaga karena setiap blok *cipher text* selalu berkaitan dengan blok sebelumnya. Setiap blok yang sama akan menghasilkan blok *cipher text* yang bervariasi dan benar-benar berbeda sehingga *cipher text* aman dari analisis frekuensi.

***Kata Kunci : CBC, XOR, Blum Blum Shub, Kriptografi***

## **ABSTRACT**

**WINDI ASMITA**

***Improving Data Security Using the XOR Algorithm in the Operation Mode of the Cipher Block Chaining (CBC) Algorithm with the Random Key Blum Blum Shub***  
**2019**

*This research was conducted to combine the XOR technique with 256-bits Cipher Block Chaining (CBC) operation mode with a 256-bits randomly generated key with the Blum Blum Shub algorithm. The resulting algorithm combination is implemented with the PHP programming language to then be tested and analyzed the results of the encryption and decryption process on the sample message used. The results showed that the processing time needed for both encryption and decryption was very fast, and the device needed to run the application was also very light, so the algorithm designed was categorized as sparing in terms of computer resources and processing time needed. The results of the security analysis also show that it takes  $3.6717 \times 10^{49}$  Million Years to break the lock using the gross force technique and the assumption of  $10^{14}$  experiments per second. The resulting cipher text is also very sensitive to changes even though only 1 bit of data can damage the entire cipher text afterwards, so the authenticity of the cipher text is also maintained because each block of cipher text is always related to the previous block. Each same block will produce a cipher text block that varies and is completely different so that the cipher text is safe from frequency analysis.*

***Keywords: CBC, XOR, Blum Blum Shub, Cryptography***

## DAFTAR GAMBAR

No.	Judul	Halaman
2.1.	Proses Enkripsi dan Dekripsi Data .....	7
2.2.	Proses Enkripsi dan Dekripsi Data dengan Algoritma Kunci Simetris ....	11
2.3.	Proses Enkripsi dan Dekripsi Data dengan Algoritma Kunci Asimetris ..	12
2.4.	Diagram Mode Pemrosesan CBC (Cipher Block Chaining).....	15
3.1.	Tahapan Penelitian .....	29
3.2.	Diagram Konteks dari Sistem .....	71
3.3.	DFD Level 1 dari Sistem .....	72
3.4.	Interface Enkripsi .....	74
3.5.	Interface Deskripsi .....	75
3.6.	Flowchart Generate Kunci 256-bits .....	77
3.7.	Flowchart Enkripsi dengan Mode CBC 256-bits .....	78
3.8.	Flowchart Dekripsi dengan Mode CBC 256-bits .....	79
4.1.	Tampilan Awal Google Chrome .....	82
4.2.	Tampilan Login dari Aplikasi .....	83
4.3.	Tampilan Halaman Enkripsi dari Aplikasi .....	83
4.4.	Tampilan Hasil Generate Key pada Aplikasi.....	85
4.5.	Tampilan Kunci yang Berhasil Disimpan oleh Aplikasi .....	85
4.6.	Tampilan Hasil Hasil Enkripsi pada Aplikasi.....	86
4.7.	Tampilan Cipher text yang berhasil Disimpan oleh Aplikasi .....	87
4.8.	Tampilan Halaman Dekripsi pada Aplikasi.....	88
4.9.	Tampilan Upload File Cipher text dan Kunci pada Aplikasi.....	88
4.10.	Tampilan Hasil Dekripsi pada Aplikasi.....	89
4.11.	Tampilan Plain text yang Berhasil Disimpan oleh Aplikasi .....	89



## DAFTAR ISI

<b>COVER .....</b>	<b>i</b>
<b>LEMBAR JUDUL.....</b>	<b>ii</b>
<b>LEMBAR PENGESAHAN.....</b>	<b>iii</b>
<b>ABSTRAK.....</b>	<b>iv</b>
<b>KATA PENGANTAR.....</b>	<b>v</b>
<b>DAFTAR ISI .....</b>	<b>viii</b>
<b>DAFTAR TABEL .....</b>	<b>x</b>
<b>DAFTAR GAMBAR.....</b>	<b>xi</b>
<b>BAB I     PENDAHULUAN</b>	
1.1. Latar Belakang .....	1
1.2. Rumusan Masalah .....	4
1.3. Batasan Masalah .....	4
1.4. Tujuan Penelitian .....	5
1.5. Manfaat Penelitian .....	5
<b>BAB II    LANDASAN TEORI</b>	
2.1. Kriptografi .....	6
2.1.1. Pengertian Kriptografi.....	6
2.1.2. Tujuan Kriptografi .....	7
2.1.3. Istilah-Istilah Pada Kriptografi .....	8
2.1.4. Algoritma Kunci Simetris .....	10
2.1.5. Algoritma Kunci Asimetris .....	12
2.2. Teknik XOR.....	13
2.3. CBC (Cipher Block Chaining).....	15
2.4. Pembangkit Bilangan Acak Semu.....	16
2.4.1. Syarat Algoritma Pembangkit Bilangan Acak Semu .....	17
2.4.2. Algoritma Blum Blum Shub (BBS).....	18
2.5. DFD (Data Flow Diagram).....	19
2.6. Flowchart .....	22
2.7. PHP .....	24
2.8. Penelitian Terdahulu .....	25

### **BAB III METODE PENELITIAN**

3.1. Tahapan Penelitian .....	29
3.2. Metode Pengumpulan Data.....	31
3.3. Analisa Sistem yang Berjalan .....	31
3.3.1. Analisis Masalah.....	32
3.3.2. Analisis Penyelesaian Masalah.....	33
3.3.3. Analisis Proses.....	37
3.3.4. Analisis Kebutuhan Sistem .....	70
3.4. Rancangan Penelitian .....	71
3.4.1. Rancangan Sistem.....	71
3.4.2. Rancangan Interface .....	73
3.4.3. Flowchart Proses.....	70

### **BAB IV HASIL DAN PEMBAHASAN**

4.1. Kebutuhan Spesifikasi Minimum Hardware dan Software .....	80
4.2. Pengujian Aplikasi dan Pembahasan .....	82
4.2.1. Enkripsi .....	84
4.2.2. Dekripsi .....	87
4.3. Analisis Keamanan.....	90
4.3.1. Analisis Keamanan Cipher text .....	90
4.3.2. Analisis Keamanan Kunci.....	90
4.4. Kelebihan dan Kekurangan Algoritma yang Dirancang .....	91
4.4.1. Kelebihan .....	91
4.4.2. Kekurangan .....	92

### **BAB V KESIMPULAN DAN SARAN**

5.1. Kesimpulan .....	93
5.2. Saran.....	94

### **DAFTAR PUSTAKA**

### **LAMPIRAN**

# **BAB I**

## **PENDAHULUAN**

### **1.1. Latar Belakang Masalah**

Keamanan pesan pada sebuah komunikasi sangat penting untuk dijaga, terutama kerahasiaan pada pesan sehingga hanya pihak-pihak yang berhak yang dapat membaca pesan tersebut. Kerahasiaan pesan dapat dilakukan dengan menyandi pesan yang dikirim dan secara langsung mengubah isi dari pesan tersebut sehingga tidak lagi memiliki arti atau makna sehingga penyadap pesan tidak dapat memahami informasi dari pesan tersebut. Teknik ini dikenal dengan istilah kriptografi.

Kriptografi akan merahasiakan informasi dengan menyandikannya ke dalam bentuk yang tidak dapat dimengerti lagi maknanya. Saat ini banyak bermunculan algoritma kriptografi yang terus dianalisis, dicoba, dan disempurnakan untuk mencari algoritma yang dianggap memenuhi standar keamanan.

Setiap algoritma memiliki tingkat keamanan yang berbeda-beda, begitu juga dengan tingkat kompleksitas yang berbeda-beda pula. Algoritma dengan tingkat keamanan yang tinggi dan dengan kompleksitas yang rendah akan sangat baik untuk diterapkan, hal ini dikarenakan proses enkripsi dan dekripsi menjadi jauh lebih cepat dan hanya memakan sumber daya komputer yang rendah.

Salah satu algoritma kriptografi yang sangat sederhana namun handal adalah algoritma enkripsi dekripsi dengan teknik XOR. Dimana setiap bit dari *plain text* akan dilakukan operasi XOR dengan setiap bit dari kunci yang dibentuk.



Teknik XOR memiliki kelebihan utama dalam hal kecepatan, dimana teknik XOR menjadi algoritma dengan kecepatan proses enkripsi dan dekripsi tercepat dibandingkan algoritma kriptografi lainnya.

Teknik XOR memiliki proses enkripsi yang sangat cepat, mudah, namun sangat andal. Keamanan pada teknik XOR dapat ditingkatkan dengan memperbanyak kunci yang dimiliki dan melakukan pengacakan terhadap kunci seacak mungkin, sehingga kemungkinan *intruder* mampu menebak kunci yang digunakan mendekati nol. Teknik XOR termasuk dalam algoritma simetris yang dapat dioperasikan dengan mode operasi blok *cipher*, dimana panjang *plain text* akan dibagi-bagi ke dalam bentuk blok-blok pesan dengan panjang tertentu. Semakin panjang blok yang digunakan maka semakin aman hasil enkripsi yang diberikan.

CBC (*Cipher Blok Chaining*) merupakan salah satu mode operasi blok *cipher*, dimana hasil enkripsi dari blok pesan sebelumnya akan digunakan sebagai *initial vektor* untuk mengacak blok pesan selanjutnya sebelum dienkripsi dengan kunci yang digunakan. Sehingga pada mode operasi CBC, *plain text* akan diproses sebanyak dua kali untuk menjamin tidak adanya pemetaan dari setiap blok *cipher text*, sehingga jika *intruder* memiliki potongan dari *plain text*, namun hal tersebut tidak akan banyak membantu.

Oleh karena itu, kekuatan teknik XOR pada mode operasi blok *cipher* CBC terletak dari panjangnya kunci yang digunakan dan pengacakan terhadap kunci. Semakin panjang kunci yang dihasilkan maka *cipher text* yang dihasilkan juga akan semakin kuat. Begitu juga semakin panjang blok yang digunakan maka

kunci yang digunakan juga semakin panjang sehingga keamanan *cipher text* yang dihasilkan juga akan semakin meningkat.

Untuk meningkatkan keacakan dari kunci yang digunakan maka dibutuhkan suatu algoritma khusus untuk membangkitkan bilangan acak. Salah satu algoritma yang dapat digunakan adalah algoritma Blum Blum Shub. Dimana algoritma ini mampu membangkitkan bilangan acak dengan tingkat pengacakan yang sangat tinggi, terutama jika umpan bilangan prima yang diberikan adalah bilangan prima dengan digit yang besar.

Untuk meningkatkan keamanan dari *cipher text*, panjang blok yang digunakan pada penelitian ini ditetapkan sepanjang 256-bits. Hal ini dilakukan karena dengan panjang blok sepanjang 256-bits, maka kemungkinan kunci yang dapat terbentuk adalah sebanyak  $1,15 \times 10^{77}$ . Sehingga blok sepanjang 256-bits dapat diasumsikan memiliki kekuatan yang sangat tinggi untuk menjaga kerahasiaan kunci, terutama dari serangan *bruto force*.

Oleh karena itu, dalam penelitian ini, algoritma teknik XOR pada mode operasi blok *cipher* CBC akan menggunakan blok sepanjang 256-bits dan menggunakan kunci acak yang dibangkitkan dengan algoritma Blum Blum Shub. Adapun laporan hasil penelitian tersebut akan disajikan dalam bentuk laporan skripsi yang berjudul: “*Meningkatkan Keamanan Data Menggunakan Operasi XOR Mode Algoritma Cipher Block Chaining (CBC) dengan Kunci Acak Blum Blum Shub*”.

## 1.2. Rumusan Masalah

Setelah diuraikan latar belakang masalah serta diambilnya identifikasi masalah yang ada, maka dapat diambil suatu rumusan masalah sebagai berikut :

1. Bagaimana meningkatkan keamanan data menggunakan operasi XOR.
2. Bagaimana menerapkan kombinasi Algoritma CBC dengan operasi XOR untuk keamanan data.
3. Bagaimana mengimplementasikan kombinasi operasi XOR mode Algoritma CBC dengan Kunci Acak Blum Blum Shub.

## 1.3. Batasan Masalah

Mengingat ruang lingkup permasalahan yang ada cukup luas, dan karena adanya keterbatasan, waktu, dana, tenaga, teori, dan agar penelitian dapat dilakukan secara lebih mendalam sehingga hasil yang dicapai dapat seperti apa yang ditujukan, maka dalam penelitian ini diberikan batasan-batasan masalah yang ada sebagai berikut :

1. Panjang blok yang digunakan sepanjang 256-bits.
2. Pesan yang dienkripsi adalah pesan teks yang digunakan merupakan kombinasi alfanumerik dan simbol yang terdaftar pada kode ASCII 8 bit.
3. Fokus pengamanan data dalam penelitian ini hanya berfokus kepada pengamanan pesan teks, tanpa memandang pesan teks tersebut akan dikirim via apa.
4. Teknik enkripsi dan dekripsi yang digunakan adalah teknik XOR
5. Mode operasi blok *cipher* yang digunakan adalah mode CBC (*Cipher Block Chaining*).

6. Kunci yang di gunakan memakai Kunci Acak Blum Blum Shub.
7. Aplikasi dibangun menggunakan bahasa pemrograman PHP.

#### **1.4. Tujuan Penelitian**

Adapun tujuan dari penelitian ini adalah :

1. Meningkatkan keamanan data menggunakan operasi XOR.
2. Mengkombinasikan Algoritma CBC dengan operasi XOR.
3. Mengimplementasikan kombinasi Operasi XOR dengan algoritma CBC dalam bentuk aplikasi.

#### **1.5. Manfaat Penelitian.**

Adapun manfaat dari penelitian ini adalah sebagai berikut :

1. Memberikan peningkatan keamanan data user menggunakan operasi XOR.
2. Tersedianya alternatif algoritma untuk keamanan data dari kombinasi algoritma CBC dengan operasi XOR..
3. Tersedianya aplikasi untuk keamanan data yang dapat digunakan oleh user dari kombinasi Operasi XOR dengan algoritma CBC.

## **BAB II**

### **LANDASAN TEORI**

#### **2.1. Kriptografi**

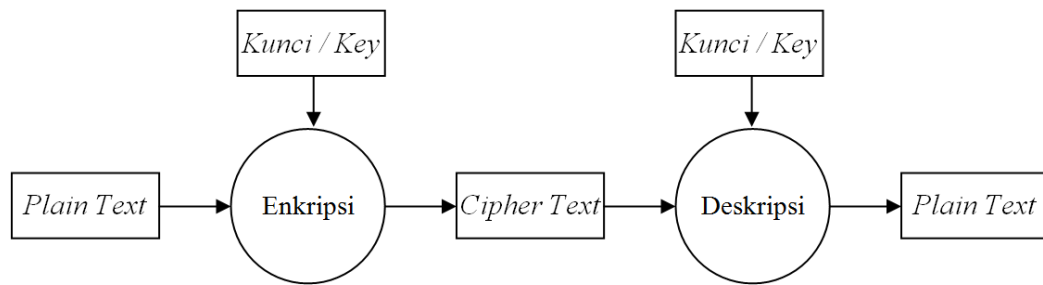
##### **2.1.1. Pengertian Kriptografi**

Kriptografi adalah ilmu mengenai teknik enkripsi dimana data diacak menggunakan suatu kunci enkripsi menjadi sesuatu yang sulit dibaca oleh seseorang yang tidak memiliki kunci dekripsi. Dekripsi menggunakan kunci dekripsi untuk mendapatkan data asli kembali. Proses enkripsi dilakukan menggunakan suatu algoritma dengan beberapa parameter. Biasanya algoritma tidak dirahasiakan, bahkan enkripsi yang mengandalkan kerahasiaan algoritma dianggap sesuatu yang tidak baik. Rahasia terletak di beberapa parameter yang digunakan, jadi kunci ditentukan oleh parameter. Parameter yang menentukan kunci dekripsi itulah yang harus dirahasiakan (parameter menjadi ekuivalen dengan kunci), Kromodimoeljo (2014).

Scheiner (2016) menyebutkan bahwa kriptografi adalah ilmu sekaligus seni untuk menjaga keamanan pesan. Sedangkan Ariyus (2017) menjelaskan bahwa kriptografi merupakan ilmu atau seni untuk menjaga keamanna pesan ketika pesan dikirimkan dari suatu tempat ke tempat yang lainnya.

Dalam arti lain, kriptografi adalah seni dan ilmu dalam mengamankan pesan. Dalam dunia kriptografi, pesan disebut *plain text* atau *clear text*. Proses untuk menyamarkan pesan dengan cara sedemikian rupa untuk menyembunyikan isi aslinya disebut enkripsi. Pesan yang telah dienkrpsi disebut *cipher text*. Proses pengembalian sebuah *cipher text* ke *plain text* disebut dekripsi.





Gambar 2.1. Proses Enkripsi dan Dekripsi Data

Sumber: Subandi (2017)

Dalam kriptografi klasik, teknik enkripsi yang digunakan adalah enkripsi simetris dimana kunci dekripsi sama dengan kunci enkripsi. Untuk kriptografi kunci publik, diperlukan teknik enkripsi asimetris dimana kunci dekripsi tidak sama dengan kunci enkripsi. Enkripsi, dekripsi dan pembuatan kunci untuk teknik enkripsi asimetris memerlukan komputasi yang lebih intensif dibandingkan enkripsi simetris, karena enkripsi asimetris menggunakan bilangan-bilangan yang sangat besar. Namun, walaupun enkripsi asimetris lebih besar dalam kompleksitas dibandingkan enkripsi simetris, kriptografi kunci publik sangat berguna untuk *key management* dan *digital signature*, Kromodimoeljo (2014).

### 2.1.2. Tujuan Kriptografi

Munir (2016), Ada empat tujuan mendasar dari kriptografi yang juga merupakan aspek keamanan informasi, yaitu:

#### 1. *Privacy/Confidentiality*

Aspek yang berhubungan dengan penjagaan isi informasi dari siapapun kecuali yang memiliki otoritas atau kunci rahasia untuk membuka informasi yang telah dienkripsi.

## 2. *Integrity*

Aspek yang berhubungan dengan penjagaan dari perubahan data secara tidak sah dan menjamin bahwa isi pesan tidak dapat diubah tanpa ijin.

## 3. *Authentication*

Aspek yang berhubungan dengan identifikasi atau pengenalan, baik secara kesatuan sistem maupun informasi itu sendiri.

## 4. *Non-repudiation*

Usaha untuk mencegah terjadinya penyangkalan terhadap pengiriman suatu informasi oleh yang mengirimkan, atau harus dapat membuktikan bahwa suatu pesan berasal dari seseorang, apabila dia menyangkal mengirim informasi tersebut.

### **2.1.3. Istilah-Istilah Pada Kriptografi**

Munir (2016) menjelaskan terdapat beberapa istilah yang dikenal pada kriptografi, yaitu:

#### 1. Algoritma Kriptografi

Sebuah algoritma yang digunakan untuk melakukan proses enkripsi terhadap *plain text* menjadi *cipher text* dan proses dekripsi terhadap *cipher text* kembali menjadi *plain text* dengan suatu aturan dan kunci tertentu. Sehingga pada suatu algoritma kriptografi terdapat tiga buah proses, yaitu: *generate key*, enkripsi, dan dekripsi.

#### 2. *Plain text*

Yaitu pesan atau data dalam bentuk aslinya dapat terbaca. *Plaintext* merupakan masukan bagi algoritma enkripsi.

### 3. *Cipher text*

*Cipher text* adalah keluaran algoritma enkripsi. *Cipher text* dapat dianggap sebagai pesan dalam bentuk tersembunyi. Algoritma enkripsi yang baik akan menghasilkan *cipher text* yang kelihatan acak. Selanjutnya digunakan istilah teks sandi sebagai sinonim kata *cipher text*

### 4. *Key* (kunci)

Sebuah kunci yang akan mempengaruhi dan menentukan hasil dari proses enkripsi dan dekripsi dengan suatu algoritma kriptografi. Kunci dapat dibangkitkan dengan suatu aturan algoritma tertentu atau dengan memilih kunci secara bebas.

Kunci terbagi menjadi dua jenis, yaitu :

#### a. Kunci Simetris

Kunci untuk melakukan proses enkripsi sama dengan kunci untuk melakukan proses dekripsi, sehingga hanya dikenal sebuah kunci untuk satu proses enkripsi dekripsi. Kunci simetris bersifat sangat rahasia, dan mutlak hanya boleh diketahui oleh pengirim dan penerima pesan.

#### b. Kunci Asimetris

Kunci untuk melakukan proses enkripsi berbeda dengan untuk melakukan proses dekripsi. Kunci untuk melakukan proses enkripsi disebut dengan *public key*. *Public key* bersifat tidak rahasia dan boleh diketahui oleh siapapun dan dapat disebarluaskan secara bebas. Kunci untuk melakukan proses dekripsi disebut sebagai *private key*. *Privatekey* bersifat sangat rahasia dan hanya boleh diketahui oleh penerima pesan

atau pembangkit kunci, bahkan pengirim pesan juga tidak diijinkan mengetahui *private key*.

#### 5. Enkripsi

Proses mengubah *plain text* menjadi *cipher text* dengan suatu algoritma kriptografi tertentu dan dengan sebuah kunci tertentu. Baik menggunakan kunci simetris atau menggunakan kunci asimetris (*public key*).

#### 6. Dekripsi

Proses mengubah *cipher text* menjadi *plain text* kembali dengan algoritma kriptografi tertentu dan dengan sebuah kunci tertentu. Baik kunci simetris atau kunci asimetris (*private key*).

#### **2.1.4. Algoritma Kunci Simetris**

Munir (2016) menjelaskan bahwa algoritma kunci simetris adalah suatu algoritma kriptografi yang melakukan proses enkripsi dan dekripsi dengan menggunakan sebuah kunci yang sama. Apabila kunci yang digunakan dalam melakukan enkripsi dan dekripsi berbeda, maka menyebabkan keluaran terakhir dari algoritma kacau, sehingga tidak berhasil mengembalikan bentuk *cipher text* ke *plain text* semula. Karena kunci ini memegang peranan yang sangat penting dalam melakukan enkripsi maupun dekripsi, maka algoritma simetris ini disebut juga dengan algoritma kunci rahasia (*secret key algorithm*).

Algoritma ini mengharuskan pengirim dan penerima pesan untuk menyetujui kunci yang akan digunakan dan keamanan dari algoritma ini tergantung dari kunci yang digunakan, sehingga kunci ini harus dirahasiakan. Jika

kunci ini disebarakan berarti semua orang dapat melakukan enkripsi dan dekripsi pesan dalam sistem tersebut.

Dalam notasi matematika, proses algoritma kunci rahasia digambarkan sebagai berikut:

$$E_k(P) = C$$

$$D_k(C) = P$$

Keterangan :

E = Fungsi enkripsi

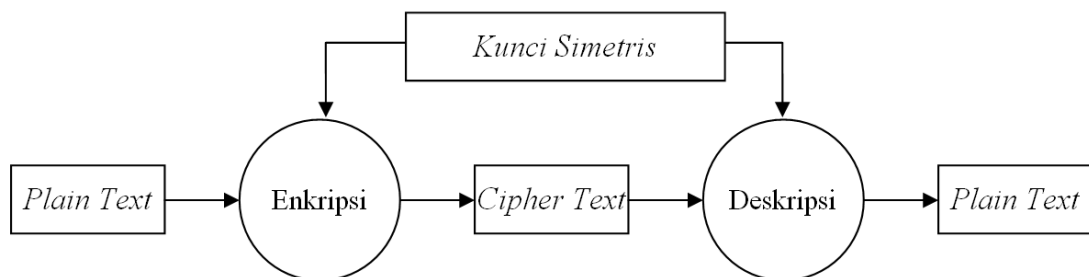
D = Fungsi dekripsi

k = Kunci simetris (sama untuk enkripsi maupun dekripsi)

C = *Cipher text*

P = Pesan (*message*) atau *plain text*.

$E_k$  dan  $D_k$  adalah fungsi enkripsi dan dekripsi yang menggunakan kunci yang sama.



Gambar 2.2. Proses Enkripsi dan Dekripsi Data dengan Algoritma Kunci Simetris  
Sumber: Subandi (2017)

Algoritma Simetris dapat dibagi menjadi dua kategori, yaitu algoritma stream (*stream algorithm*) atau *stream ciphers*, yang beroperasi pada setiap bit

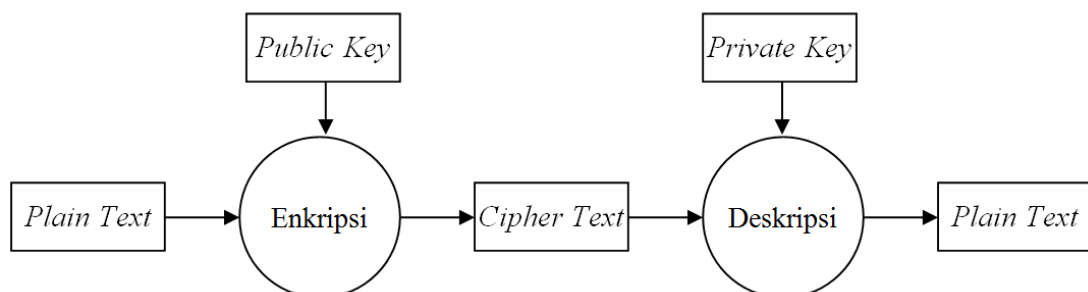


dari *plain text*, dan algoritma blok (*block algorithms*) atau *cipher* blok (*block ciphers*), yang beroperasi pada kelompok-kelompok bit.

### 2.1.5. Algoritma Kunci Asimetris

Munir (2016) menjelaskan bahwa algoritma kunci publik (disebut juga algoritma kunci asimetris) dirancang sedemikian rupa sehingga kunci yang digunakan untuk enkripsi berbeda dari kunci yang digunakan untuk dekripsi. Selanjutnya, kunci dekripsi tidak dapat dihitung dengan kunci enkripsi (setidaknya dengan proses yang sangat sulit dan atau dengan waktu yang cukup lama).

Algoritma ini disebut kunci publik karena kunci enkripsi dapat diberitahukan ke publik: Orang asing dapat menggunakan kunci enkripsi untuk mengenkripsi pesan, namun hanya orang-orang tertentu saja dengan kunci dekripsi yang sesuai yang dapat mendekripsi pesan. Dalam sistem ini, kunci enkripsi sering disebut sebagai *public key*, dan kunci dekripsi sering disebut *private key*. *Private key* terkadang disebut kunci rahasia, tetapi untuk menghindari kerancuan dengan algoritma simetris, istilah tersebut tidak digunakan, Schneier (2016).



Gambar 2.3. Proses Enkripsi dan Dekripsi Data dengan Algoritma Kunci Asimetris

Sumber: Subandi (2017)

Enkripsi menggunakan kunci publik  $K$  disimbolkan :

$$E_{\text{public}}(P) = C$$

Walaupun *publickey* dan *privatekey* berbeda, dekripsi dengan *privatekey* yang cocok disimbolkan dengan:

$$D_{\text{private}}(C) = P$$

Keterangan :

$E$  = Fungsi enkripsi

$D$  = Fungsi dekripsi

*Public* = *Public Key*

*Private* = *Private Key*

$C$  = *Cipher text*

$P$  = Pesan (*message*) atau *plain text*.

## 2.2. Teknik XOR

Salah satu algoritma kriptografi simetri yang cukup sederhana adalah algoritma kriptografi XOR. Sesuai dengan namanya, algoritma ini melakukan enkripsi dan dekripsi terhadap sebuah informasi dengan menggunakan kunci tunggal dan operasi bit XOR. Tabel logika dari operasi XOR adalah sebagai berikut: Munir (2016)

Tabel 2.1. Tabel Logika Operasi XOR

A	B	$A \oplus B$
0	0	0
0	1	1
1	0	1
1	1	0

Sumber: Munir (2016)

Beberapa keunggulan operasi XOR dalam melakukan enkripsi adalah:

Munir (2016)

1. Bersifat *Closure*

Artinya operasi XOR terhadap suatu *plain text* yang panjangnya nbit akan menghasilkan *ciphertext* yang panjangnya juga n-bit.

2. Bersifat Asosiatif.

$$A \oplus (B \oplus C) = (A \oplus B) \oplus C$$

3. Bersifat Komutatif

$$A \oplus B = B \oplus A$$

Dengan demikian algoritma kriptografi XOR memiliki sifat *commutative encryption*. Perubahan urutan proses enkripsi dan dekripsi ganda terhadap sebuah *plain text* dengan menggunakan kunci yang sesuai akan menghasilkan *plain text* yang sama:  $D_{k2}(D_{k1}(E_{k2}(E_{k1}(P))))=D_{k1}(D_{k2}(E_{k2}(E_{k1}(P))))=P$ .

Sebagai contoh akan dilakukan enkripsi dan dekripsi sebanyak masing-masing dua kali terhadap *plain text*: 10101110,  $k-1$ (kunci-1): 11101110 dan  $k-2$  (kunci-2): 01101001. Meskipun urutan proses enkripsi-dekripsi yang akan dilakukan berbeda, operasi XOR akan menghasilkan *plain text* yang sama bila jumlah enkripsi dan dekripsi yang dilakukan jumlahnya sama dan menggunakan kunci yang bersesuaian.

Tabel 2.2. Sifat Commutative Encryption pada XOR

$D_{k1}(D_{k2}(E_{k1}(E_{k2}(P))))$	$D_{k2}(E_{k2}(D_{k1}(E_{k1}(P))))$
P:10101110	P:10101110
$C_1 = E_{k2}(P):11000111$	$C_1 = E_{k1}(P):01000000$
$C_2 = E_{k1}(C_1):00101001$	$P = D_{k1}(C_1):10101110$
$C_1 = D_{k2}(C_2):01000000$	$C_2 = E_{k2}(P):11000111$
$P = D_{k1}(C_1):10101110$	$P = D_{k2}(C_2):10101110$

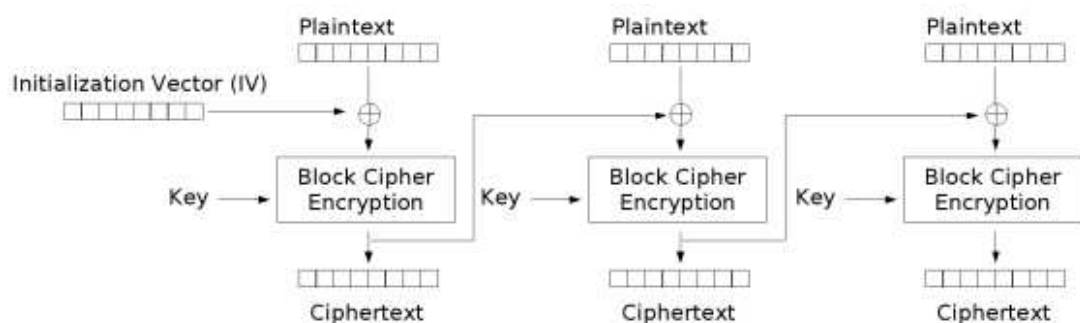
Sumber: Munir (2016)

### 2.3. CBC (*Cipher Block Chaining*)

Mode *Cipher Block Chaining* (CBC) menerapkan mekanisme umpan-balik (*feedback*) pada sebuah blok, yang dalam hal ini hasil enkripsi blok sebelumnya diumpan-balikkan ke dalam enkripsi blok yang sedang diproses. Caranya, blok *plain text* yang sedang diproses di-XOR-kan terlebih dahulu dengan blok *cipher text* hasil enkripsi sebelumnya, selanjutnya hasil peng-XOR-an ini masuk ke dalam fungsi enkripsi, Pascal (2017).

Pada mode *Cipher Block Chaining* (CBC), setiap blok *cipher text* bergantung tidak hanya pada blok *plain text*nya tetapi juga pada seluruh blok *plain text* sebelumnya. Dekripsi dilakukan dengan memasukkan blok *cipher text* yang sedang diproses ke fungsi dekripsi, kemudian meng-XOR-kan hasilnya dengan blok *cipher text* sebelumnya. Dalam hal ini, Blok *cipher text* sebelumnya berfungsi sebagai umpan-maju (*feed forward*) pada akhir proses dekripsi.

Pada Gambar berikut memperlihatkan skema proses Enkripsi dan Dekripsi pada *Cipher Block Chaining* (CBC).



Gambar 2.4. Diagram Mode Pemrosesan CBC (*Cipher Block Chaining*)

Sumber:Pascal (2017)

Secara matematis, enkripsi dengan *Cipher Block Chaining* (CBC) dinyatakan pada persamaan:

$$C_i = E_k(P_i \oplus C_{i-1})$$

Pada metode CBC (*Cipher Block Chaining*) blok *plain text* pertama menggunakan  $C_0$  sebagai vektor awal (initialization vector atau IV) yang didefinisikan dengan bit 0 sepanjang blok yang digunakan. Blok-blok *plain text* yang identik dienkripsi menjadi blok-blok *cipher text* yang berbeda hanya jika blok-blok *plain text* sebelumnya berbeda. Jika blok-blok *plain text* sebelumnya ada yang sama, maka ada kemungkinan *cipher text*nya sama. Untuk mencegah hal ini, maka digunakan IV yang merupakan data acak sebagai blok pertama. IV tidak mempunyai makna, hanya digunakan untuk membuat tiap blok *cipher text* menjadi unik. Keuntungan Mode *Cipher Block Chaining* (CBC) pesan menjadi jauh lebih aman untuk dideteksi kuncinya karena kunci tiap blok berbeda beda bergantung dari *plain text* sebelumnya. Terlihat bahwa dengan menggunakan mode CBC, blok *plain text* yang sama dienkripsikan menjadi dua blok *cipher text* yang berbeda. Setelah proses enkripsi dilakukan, maka proses selanjutnya adalah merubah ciphertext menjadi *plain text*, proses ini disebut dengan proses dekripsi. Proses dekripsi merupakan proses kebalikan dari proses enkripsi.

#### **2.4. Pembangkit Bilangan Acak Semu**

Pembangkitan bilangan acak sangat dibutuhkan oleh berbagai bidang keilmuan. Namun dua bidang keilmuan yang sangat bergantung pada pembangkitan bilangan acak adalah bidang statistik dan kriptografi. Pada bidang statistik, bilangan acak dibutuhkan untuk menghasilkan suatu analisa statistik, untuk itu bilangan acak harus benar-benar acak. Bila bilangan acak memiliki sifat-sifat tertentu karena hasil perhitungan aritmatika tertentu, misal: berulang, maka

hasil analisis statistiuk yang dihasilkan tidak akurat dan kurang dapat dipercaya. Pada bidang kriptografi, bilangan acak dibutuhkan untuk menghasilkan berbagai hal, seperti menghasilkan kata kunci yang baik, menghasilkan kunci dan padanannya pada algoritma kriptografi asimetrik/publik, membangkitkan vektor inisialisasi pada hampir seluruh algoritma kriptografi dan lainnya, Lietara (2016).

#### **2.4.1. Syarat Algoritma Pembangkit Bilangan Acak Semu**

Algoritma pembangkit bilangan acak harus setidaknya memenuhi syarat-syarat yang telah ditentukan pada *Request for Commands 1750 (Randomness Recommendations for Security)*. Ringkasan dari RFC 1750 adalah sebagai berikut: Lietara (2016)

1. Tidak dapat diprediksi. Suatu algoritma pembangkit bilangan acak harus bersifat tidak dapat diprediksi hasil keluarannya. Sebab dengan dapat memprediksi keluaran suatu algoritma, maka dengan tidak langsung dapat pula mengetahui kelauran dari fungsi kriptografi secara keseluruhan.
2. Untuk algoritma yang berdasarkan pada perangkat keras maka harus tahan akan pembelokan. Yaitu bila perangkat keras dimanipulasi, maka algoritma dapat tahan terhadap manipulasi sumber bilangan tersebut hingga level tertentu yang tidak mungkin lagi dapat ditahan.
3. Untuk algoritma yang berdasarkan perangkat lunak/aritmatika, maka harus memiliki fungsi penggabungan dan pengacakan. Yang berguna untuk mencegah penyerangan terhadap algoritma secara keseluruhan.



### 2.4.2. Algoritma Blum Blum Shub (BBS)

Blum-blum shub cukup terkenal sebagai sebuah algoritma pembangkit bilangan acak yang baik dikarenakan kemangkusannya dan kesederhanaannya. BBS dibuat pada tahun 1986 oleh tiga orang: Lenore Blum, Manuel Blum, dan Michael Shub, Lietara (2016).

Kromodimoeljo (2014) menjelaskan bahwa algoritma Blum Blum Shub merupakan salah satu algoritma pembangkit bilangan acak semu yang kuat, tidak mudah diprediksi, dan cukup ringan walaupun menggunakan perasi perpangkatan.

Pada penelitian ini, kunci enkripsi-dekripsi algoritma *one time pad* yang digunakan serta kunci perotasian *cipher text* dibangkitkan dengan pembangkit bilangan acak *cryptographically secure pseudorandom generator* (CSPRNG) menggunakan teori bilangan *Blum-Blum Shub*. Algoritma pembangkitan bilangan acak BBS adalah sebagai berikut: Munir (2016)

1. Pilih dua buah bilangan prima rahasia  $p$  dan  $q$ , yang masing-masing kongruen 3 modulo 4 atau  $3 = x \pmod{4}$  (dalam praktek bilangan prima yang digunakan cukup besar).
2. Kalikan keduanya menjadi  $n = pq$ . Bilangan  $n$  ini disebut bilangan bulat Blum.
3. Pilih bilangan acak lain  $s$ , sebagai umpan sedemikian sehingga:
  - (i)  $2 \leq s \leq n$
  - (ii)  $s$  dan  $n$  relatif prima
 Kemudian hitung  $x_0 = s^2 \pmod{n}$

4. Barisan bit acak dihasilkan dengan melakukan iterasi berikut sepanjang yang diinginkan.

(iii) hitung  $x_i = x_{i-1}^2 \bmod n$

(iv)  $z_i = \text{bit LSB dari } x_i$

Barisan bit acak yang dihasilkan adalah  $z_1, z_2, z_3, \dots$

Yang menarik dari algoritma pembangkit bilangan acak ini adalah bahwa tidak perlu melakukan iterasi untuk mendapatkan bilangan acak jika  $p$  dan  $q$  diketahui, sebab  $x_i$  dapat dihitung secara langsung. Keamanan BBS terletak pada sulitnya memfaktorkan. Nilai  $n$  tidak perlu rahasia dan dapat diumumkan kepada publik. BBS tidak dapat diprediksi dari arah kiri dan tidak dapat diprediksi dari arah kanan, yaitu berarti jika diberikan barisan bit yang dihasilkan oleh BBS, kriptanalis tidak dapat menganalisa barisan bit sebelumnya dan barisan bit sesudahnya dalam urutan bilangan acak yang dihasilkan.

## 2.5. DFD (*Data Flow Diagram*)

Ladjamudin (2018) menyebutkan ”*Data Flow Diagram* merupakan model dari sistem untuk menggambarkan pembagian sistem ke modul yang lebih kecil. DFD menggunakan notasi–notasi (simbol–simbol) untuk menggambarkan arus data”.

DFD sering digunakan untuk menggambarkan suatu sistem yang telah ada atau sistem yang akan digambarkan secara logika tanpa mempertimbangkan lingkungan fisik dimana data tersebut mengalir.

*Data Flow Diagram* memiliki dua kegunaan, yaitu memberikan indikasi bagaimana data ditransformasikan di dalam sistem (dari sejak masuk hingga

menghasilkan keluaran), dan menunjukkan fungsi dan subfungsi yang mentransformasikan aliran data.

Beberapa hal yang harus diperhatikan dalam membuat DFD antara lain :

1. DFD dapat terdiri dari beberapa level. Level 0 merupakan level tertinggi. Level 1 adalah penjabaran dari level 0, level 2 adalah penjabaran dari level 1, dst. Semakin rendah levelnya, fungsi yang digambarkan akan semakin rinci.
2. DFD level 0 hanya memiliki satu *bubble* (proses)
3. Setiap proses harus diberi nomor: level no-urut.
4. Masukan dan keluaran utama harus diperhatikan dengan seksama.
5. Proses *refinement* dimulai dengan menentukan kandidat proses, objek data, dan penyimpanan data yang akan direpresentasikan di level berikutnya.
6. Setiap panah (*arrow*) harus diberi nama yang mencerminkan arti panah tersebut.
7. Aliran informasi harus dijaga dari satu level ke level lainnya.
8. Lakukan proses dekomposisi satu per satu (per *bubble*).
9. Tuliskan sebuah P-SPEC (spesifikasi singkat dalam bahasa Inggris atau notasi algoritma) untuk setiap *bubble* di DFD final.

Di dalam suatu data flow diagram terdapat beberapa elemen, yaitu:

1. *Process*

*Process* merupakan suatu aktifitas atau fungsi yang menggambarkan secara spesifik tujuan dari proses bisnis.

## 2. Data Flow

*Data flow* menunjukkan suatu aliran dari data. Aliran data ini biasanya menghubungkan *process* dengan *process*, *process* dengan *data store*, juga menghubungkan *external entity* dengan *process*.


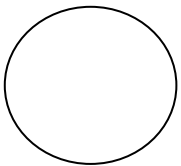
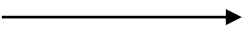

## 3. Data Store

*Data store* merupakan kumpulan dari data yang disimpan dengan beberapa cara. Suatu data bisa mengalir masuk kedalam *data store* juga dapat keluar dari *data store*.

## 4. External Entity

*External entity* merupakan gambaran dari orang, *user*, ataupun organisasi yang berinteraksi secara langsung dengan sistem.

Tabel 2.3. Simbol-Simbol Elemen DFD Veri Yourdan De Marco

No	Simbol	Nama Simbol	Fungsi
1.		<i>Entity External</i> (Kesatuan Luar)	Data harus selalu berasal dari suatu tempat dan harus selalu dikirim ke suatu <i>storage</i>
2.		<i>Process</i> <i>/Atribut</i>	Mengubah <i>input</i> menjadi <i>output</i> Data harus selalu diproses dengan cara tertentu untuk menghasilkan fungsi sistem.
3.		<i>Data Flow</i> (Aliran Data)	Data mengalir melalui sistem dimulai dengan sebagian <i>input</i> dan diubah atau diproses menjadi <i>output</i> .
4.		<i>Data Storage</i> (Penyimpanan Data)	Untuk menggambarkan simpanan data yang dapat berupa <i>file</i> atau <i>database</i> .

Sumber: Ladjamudin (2018)

## 2.6. Flowchart

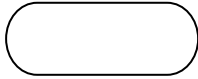
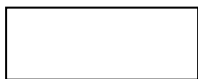
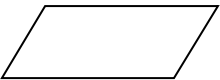
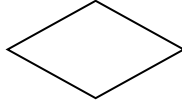

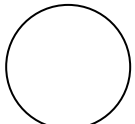
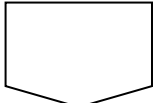
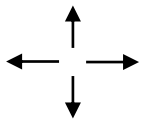

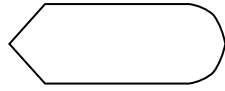
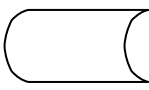
*Flowchart* yang juga disebut dengan diagram alir merupakan kumpulan simbol-simbol atau skema yang menunjukkan/menggambarkan rangkaian kegiatan-kegiatan program dari awal hingga akhir. *Flowchart* ini merupakan penggambaran dari urutan langkah-langkah pekerjaan dari suatu algoritma.

Menurut Jogiyanto (2015) “*flowchart* adalah sekumpulan simbol-simbol yang menunjukkan atau menggambarkan rangkaian kegiatan-kegiatan program dari awal hingga akhir atau suatu bagan yang menggambarkan alir logika dari data yang akan diproses dalam suatu program dari awal sampai akhir bagan alir terdiri dari simbol-simbol yang mewakili fungsi-fungsi langkah program dan garis alir (*flowlines*) menunjukkan alir terdiri dari simbol-simbol yang akan dikerjakan”.

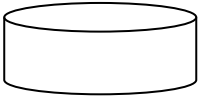
Ladjamudin (2018) menyatakan “*flowchart* adalah bagan-bagan yang memiliki arus yang menggambarkan langkah-langkah penyelesaian suatu masalah”.

Tujuan utama pembuatan *flowchart* ini adalah untuk menggambarkan suatu tahapan penyelesaian masalah sederhana, terurai, rapi dan jelas. *Flowchart* atau diagram alir merupakan kumpulan simbol-simbol atau skema yang menunjukkan/menggambarkan rangkaian kegiatan-kegiatan program dari awal hingga akhir. *Flowchart* ini merupakan penggambaran dari urutan langkah-langkah pekerjaan dari suatu algoritma. Adapun simbol-simbol *flowchart* lihat pada tabel sebagai berikut:

Tabel 2.4. Simbol-Simbol Flowchart

NO	SIMBOL	Nama Simbol	FUNGSI
1.		<i>Terminal</i>	untuk memulai atau mengakhiri suatu program
2.		Proses	suatu simbol yang menunjukkan setiap pengolahan yang dilakukan.
3.		<i>Input-Output</i>	untuk memasukkan menunjukkan hasil dari suatu proses
4.		<i>Decision</i>	suatu kondisi yang akan menghasilkan beberapa kemungkinan jawaban atau pilihan
5.		<i>Preparation</i>	suatu simbol yang menyediakan tempat pengolahan
6.		<i>Connector</i>	suatu prosedur penghubung yang akan masuk atau keluar melalui simbol ini dalam lembar yang sama
7.		<i>Off-Page Connector</i>	merupakan simbol masuk atau keluarannya suatu prosedur pada lembaran kertas lainnya
8.		<i>Arus/Flow</i>	dari pada prosedur yang dapat dilakukan atas ke bawah dari bawah ke atas, ke atas dari kiri ke kanan ataupun dari kanan ke kiri
9.		<i>Predefined Process</i>	untuk menyatakan sekumpulan langkah proses yang ditulis sebagai prosedur
10.		<i>Display</i>	Simbol untuk <i>output</i> , yang ditunjukkan ke suatu <i>device</i> , seperti <i>printer</i> , dan sebagainya
11.		<i>Storage Data</i>	Penyimpanan <i>file</i> secara sementara



NO	SIMBOL	Nama Simbol	FUNGSI
12		<i>Magnetic Disk</i>	Menunjukkan <i>input / Output Harddisk</i> (media penyimpanan)

Sumber: Ladjamudin (2018)

## 2.7. PHP

PHP adalah bahasa pemrograman *script server-side* yang didesain untuk pengembangan *web*. Selain itu, PHP juga bisa digunakan sebagai bahasa pemrograman umum. MADCOMS (2015)

MADCOMS (2015) menerangkan bahwa pada awalnya PHP merupakan singkatan dari Personal Home Page (Situs personal). PHP pertama kali dibuat oleh Rasmus Lerdorf pada tahun 1995. Pada waktu itu PHP masih bernama Form Interpreted (FI), yang wujudnya berupa sekumpulan skrip yang digunakan untuk mengolah data formulir dari *web*. Selanjutnya Rasmus merilis kode sumber tersebut untuk umum dan menamakannya PHP/FI. Dengan perilsan kode sumber ini menjadi sumber terbuka, maka banyak pemrogram yang tertarik untuk ikut mengembangkan PHP.

PHP di kembangkan pada tahun 1995 oleh Rasmus Lerdorf, dan sekarang dikelola oleh *The PHP Group*. Situs resmi PHP beralamat di <http://www.php.net>. PHP disebut bahasa pemrograman *server side* karena PHP diproses pada komputer *server*. Hal ini berbeda dibandingkan dengan bahasa pemrograman *client-side* seperti JavaScript yang diproses pada *web browser (client)*.

Pada awalnya PHP merupakan singkatan dari *Personal Home Page*. Sesuai dengan namanya, PHP digunakan untuk membuat *website* pribadi. Dalam beberapa tahun perkembangannya, PHP menjelma menjadi bahasa pemrograman

*web* yang *powerful* dan tidak hanya digunakan untuk membuat halaman *web* sederhana, tetapi juga *website* populer yang digunakan oleh jutaan orang seperti *wikipedia*, *wordpress*, *joomla*, dan lain sebagainya.

Saat ini PHP adalah singkatan dari **PHP: Hypertext Preprocessor**, sebuah kepanjangan rekursif, yakni permainan kata dimana kepanjangannya terdiri dari singkatan itu sendiri: **PHP: Hypertext Preprocessor**. PHP dapat digunakan dengan gratis dan bersifat *Open Source*. PHP dirilis dalam lisensi *PHP License*, sedikit berbeda dengan lisensi *GNU General Public License (GPL)* yang biasa digunakan untuk proyek *Open Source*. Kemudahan dan kepopuleran PHP sudah menjadi standar bagi programmer *web* di seluruh dunia. Menurut *wikipedia* pada februari 2014, sekitar 82% dari *web server* di dunia menggunakan PHP. PHP juga menjadi dasar dari *aplikasi CMS (Content Management System)* populer seperti *Joomla*, *Drupal*, dan *WordPress*.

## 2.8. Penelitian Terdahulu

Berikut adalah beberapa penelitian yang terkait dengan penelitian ini:

Tabel 2.5. Penelitian Terkait

No	Nama	Tahun	Judul	Hasil Penelitian	Hubungan dengan Penelitian Saai Ini
1	D. Sravan Kumar, C. H. Suneetha, dan A.Chandrasekhar	2016	<i>A Block Cipher Using Rotation and Logical XOR Operations</i>	Metode baru enkripsi data dalam blok menggunakan operasi Rotasi dan <i>Logical XOR</i> mampu meningkatkan kekuatan dari cipher text dari berbagai serangan-serangan kriptanalisis	Penelitian ini menganalisa penggunaan operasi XOR untuk keamanan data. Penelitian ini penting bagi penelitian yang sedang dilakukan untuk memahami penggunaan dari operasi XOR dalam melakukan enkripsi dan dekripsi data
2	Sridevi	2014	<i>Construction of Stream Ciphers</i>	Banyak stream cipher lebih unggul daripada	Penelitian ini melakukan analisa terhadap

No	Nama	Tahun	Judul	Hasil Penelitian	Hubungan dengan Penelitian Saai Ini
			<i>from Block Ciphers and their Security</i>	memblokir cipher terkait dengan propagasi kesalahan. Karenanya, membangun stream cipher dari cipher blok dapat menjadi alternatif yang berguna untuk stream cipher lainnya.	keamanan cari mode operasi blok cipher dan merancang algoritma blok cipher dan menganalisa keamanannya. Penelitian ini penting bagi penelitian yang saat ini sedang dilakukan untuk memahami keamanan dari algoritma blok cipher khususnya mode operasi <i>Electronic Code Book (ECB)</i>
3	Anita Dashti, Hashem Alvandi Kheradmand, dan Mohammad Davarpanah Jazi	2016	<i>Comparison Of Three Modes Of Cryptography Operation For Providing Security and Privacy Based on Important Factors</i>	Dalam makalah ini tiga mode klasik operasi CBC, CFB dan OFB telah diusulkan dalam literatur untuk perbandingan. Fitur perbandingan yang digunakan adalah ukuran blok, throughput, akses acak, propagasi kesalahan, paralelisasi, jenis sandi, dan plaintext yang identik.	Penelitian ini melakukan analisa dan perbandingan dari beberapa mode operasi blok cipher yang sangat berguna bagi penulis untuk memahami skema kerja mode operasi blok cipher yang akan digunakan pada penelitian ini.
4	Gulshan Kumar, Mritunjay Rai, and Gang-soo Lee	2015	<i>Implementation of Cipher Block Chaining in Wireless Sensor Networks for Security Enhancement</i>	Kombinasi DES dan Blowfish dalam mode CBC memberikan mekanisme yang efektif untuk peningkatan keamanan yang memberikan kerahasiaan dan otentikasi data yang lebih tinggi.	Penelitian ini menunjukkan efektifitas dari kombinasi algoritma DES dan Blowfish pada mode operasi CBC untuk peningkatan keamanan yang menyediakan kerahasiaan dan otentifikasi data yang tinggi. Penelitian menunjukkan konsep mode CBC dengan menggunakan Algoritma DES dan Blowfish
5	Dewi Rosmala, Riki Aprian	2015	Implementasi Mode Operasi Cipher Block Chaining (CBC) Pada Pengamanan Data	Perangkat lunak pengamanan data menggunakan metode <i>Cipher Block Chaining (CBC)</i> dapat merahasiakan data dengan cara merubah data seperti file teks, Rich Text Format, file	Penelitian ini menunjukkan implementasi teknik XOR dengan mode CBC pada perangkat lunak untuk mengamankan data teks. Penelitian ini menunjukkan bagaimana

No	Nama	Tahun	Judul	Hasil Penelitian	Hubungan dengan Penelitian Saai Ini
				dokumen menjadi ciphertext yang tidak dapat dimengerti.	implementasi CBC pada aplikasi yang berguna pada penelitian yang sedang dilakukan
6	Siska Anraenia, Herdiantia, Mursyid	2016	<i>Hybrid Methods of Ciphertext and RSA Cryptographic Algorithm Using Classical Vigenère</i>	Penggabungan kedua metode ini menghasilkan ciphertext yang lebih kuat dan sulit dipecahkan.	Penelitian ini menunjukkan kombinasi dari algoritma RSA dan Vigenere Cipher, dimana cipher text hasil Vigenere cipher akan dienkripsi kembali dengan algoritma RSA untuk menjadi cipher text yang lebih kuat dan lebih sulit untuk dipecahkan. Penelitian ini menunjukkan teknik dari kombinasi sehingga sangat berguna dalam penelitian yang sedang dilakukan.
7	Arif Kurnia Rachman	2015	Perbandingan Mode Chiper Electronic Code Book dan Chiper Block Chaining Dalam Pengamanan Data	Dalam ECB muncul dari fakta bahwa karena blok plainteks yang sama selalu dienkripsi menjadi blok cipherteks yang sama, sedangkan mode CBC blok cipherteks bergantung tidak hanya pada blok plainteksnya tetapi juga pada seluruh blok plainteks sebelumnya	Penelitian ini menunjukkan bahwa pada ECB blok plainteks yang sama selalu dienkripsi menjadi blok cipherteks yang sama, sedangkan mode CBC blok cipherteks bergantung tidak hanya pada blok plainteksnya tetapi juga pada seluruh blok plainteks sebelumnya. Penelitian ini penting bagi penelitian yang sedang dilakukan. Penelitian ini menunjukkan kekurangan dan kelebihan dari ECB dan CBC
8	Abdessalem Abidi, Qianxue Wang, Belgacem bouallègue, Mohsen Machhout and Christophe Gyeux	2016	<i>Quantitative Evaluation of Chaotic CBC Mode of Operation</i>	<i>CBC Mode</i> memiliki pengacakan <i>plain text</i> menjadi <i>cipher text</i> dengan sangat baik, namun kerusakan sebuah bit dari <i>cipher text</i> akan menghancurkan rangkaian <i>cipher text</i> selanjutnya sehingga	Penelitian ini mengevaluasi kesensitifan dan expansivitas dari mode CBC untuk memahami ketahanan mode ini dari Chaos. Penelitian ini membantu penulis memahami tingkat keamanan dan

No	Nama	Tahun	Judul	Hasil Penelitian	Hubungan dengan Penelitian Saai Ini
				<i>plain text</i> tidak dapat dikembalikan.	ketahanan dari mode CBC
9	Andreas Parry Lietara	2019	Analisis & Perbandingan Blum Blum Shub dan Inversive Congruential Generator Beserta Implementasinya	Makalah ini akan membahas mengenai dua algoritma pembangkit bilangan acak, yaitu Blum Blum Shub dan <i>Inversive Congruential Generator</i> , baik itu cara algoritmanya, cara kerjanya serta perbandingan kedua algoritma tersebut. Dimana Blum Blum Shub lebih mangkus dan lebih aman dari pada <i>Inversive Congruential Generator</i>	Penelitian ini membandingkan dua buah algoritma bilangan acak, yaitu <i>Blum Blum Shub</i> dan <i>Inversive Congruential Generator</i> . Dari segi keamanan dan keceptan prosesnya. Adanya penelitian ini, sangat membantu penulis untuk menentukan algoritma pembangkit bilangan acak yang mana yang lebih efektif dan efisien untuk digunakan dalam penelitian yang dilakukan saat ini.
10	Muhammad Barja Sanjaya	2017	Perancangan dan Implementasi Random Number Blum Blum Shub pada Dynamic Cell Spreading untuk Pengamanan Berkas	Algoritma Blum Blum Shub mampu meningkatkan keacakan pada parameter-parameter yang digunakan algoritma Dynamic Cell Spreading	Penelitian ini menunjukkan betapa kuatnya algoritma Blum Blum Shub dan perannya dalam meningkatkan keacakan suatu bilangan yang dibutuhkan pada pembangkitan kunci pada algoritma kriptografi.

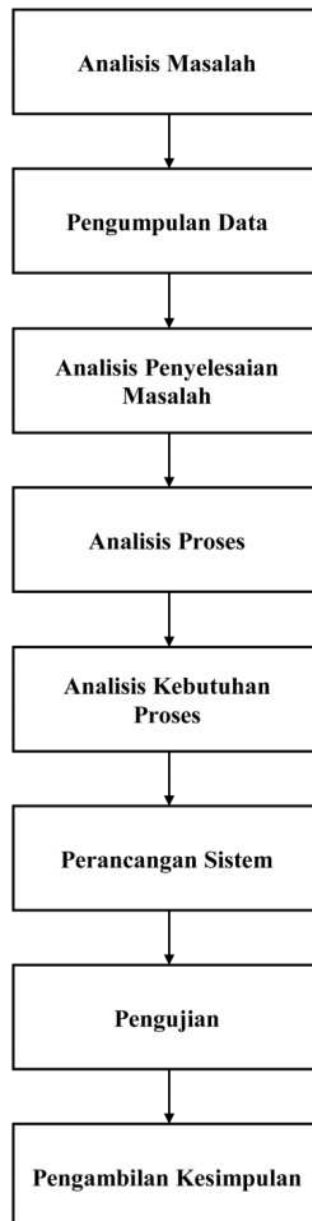
Sumber: Berbagai Data yang Dikumpulkan (2019)

## **BAB III**

### **METODE PENELITIAN**

#### **3.1. Tahapan Penelitian**

Penelitian ini dilakukan dengan beberapa tahapan yang dapat dilihat pada bagan berikut:



*Gambar 3.1. Tahapan Penelitian*

*Sumber: Oleh Penulis (2019)*



Penjelasan dari masing-masing tahapan pada bagan di atas adalah sebagai berikut:

### **1. Analisis Masalah**

Tahap ini dilakukan untuk menganalisis masalah yang akan menjadi fokus penelitian untuk kemudian diteliti dan mendapatkan penyelesaian dari masalah.

### **2. Pengumpulan Data**

Teknik pengumpulan data yang dilakukan untuk mendukung penelitian yang sedang dilakukan.

### **3. Analisis Penyelesaian Masalah**

Analisis penyelesaian masalah dari masalah yang menjadi fokus penelitian berdasarkan berbagai data dan teori yang berhasil dikumpulkan sehingga terbentuk suatu bentuk penyelesaian masalah.

### **4. Analisis Proses**

Analisis proses dari penyelesaian masalah yang diusulkan yang terdiri dari tahap pembentukan kunci, tahap enkripsi, dan tahap dekripsi

### **5. Analisis Kebutuhan Sistem**

Analisis terhadap berbagai kebutuhan untuk membangun sistem dalam menerapkan penyelesaian masalah yang diusulkan agar dapat diterapkan dalam bentuk program.

## **6. Perancangan Sistem**

Melakukan perancangan untuk membangun sistem yang terdiri dari rancangan sistem, rancangan database, dan rancangan *interface*.

## **7. Pengujian**

Melakukan pengujian terhadap sistem yang berhasil dibangun. Pengujian dimaksudnya untuk membuktikan apakah penyelesaian masalah yang diusulkan mampu menyelesaikan masalah yang menjadi fokus penelitian.

## **8. Pengambilan Kesimpulan**

Mengambil kesimpulan berdasarkan pengujian yang berhasil dilakukan.

### **3.2. Metode Pengumpulan Data**

Penelitian ini berjenis penelitian laboratorium. Pengumpulan data dilakukan dengan mengumpulkan berbagai referensi melalui buku, jurnal nasional, dan jurnal internasional. Penelitian dilakukan dengan melakukan eksperimen komputer dengan menggunakan tools bahasa pemrograman.

### **3.3. Analisa Sistem yang Berjalan**

Meningkatnya kebutuhan akan dan informasi menyebabkan terjadinya perpindahan informasi dalam jumlah besar dan waktu yang cepat. Komunikasi yang bersifat private seharusnya hanya pengirim dan penerima informasi saja yang mengetahui data tersebut. Namun pada kenyataannya banyak terjadi penyadapan yang bertujuan menyalin atau membaca data tersebut tanpa izin. Hal ini sangat berbahaya jika data tersebut merupakan data yang penting dan dapat merugikan pihak pengirim atau penerima data bahkan pihak lain.

Banyak masyarakat yang belum menyadari tidak amannya data yang dikirimkan dari kemungkinan disadap dan disalahgunakan, sehingga membuat masyarakat dengan ceroboh mengirimkan data mereka begitu saja tanpa melakukan pengamanan. Pengiriman data tanpa dilakukan pengamanan terlebih dahulu sangat rentan untuk disadap dimana kemudian data tersebut dapat disalahgunakan oleh pihak lain. Terlebih saat ini data dikirim melalui jaringan dunia dengan memanfaatkan teknologi informasi dan komunikasi, sehingga segala data dapat dengan mudah disadap dari tempat yang sangat jauh atau bahkan di belahan dunia yang lain.

### **3.3.1. Analisis Masalah**

Pendistribusian data antar pengguna membutuhkan jaminan keamanan dimana data yang dikirim tidak dapat dibaca oleh orang lain, sehingga dibutuhkan suatu teknik untuk mengamankannya. Pengiriman data tanpa pengamanan sangat rentan disadap oleh orang lain, sehingga kerahasiaan pesan yang dikirim oleh pengirim pesan menjadi terancam. Jika pesan tersebut merupakan pesan-pesan yang sangat rahasia, maka sangat berbahaya bila pesan tersebut sampai terbaca oleh orang lain yang tidak berhak. Untuk itu, dibutuhkan sebuah teknik pengamanan data yang dapat menjaga kerahasiaan pesan yang dikirim, baik menjaga kerahasiaan pesan saat ditransmisikan atau saat pesan telah diterima dan disimpan oleh penerima pesan.

Kerahasiaan menjadi hal yang sangat penting dalam komunikasi atau pertukaran data. Di era teknologi informasi saat ini, informasi memegang peranan

yang sangat penting. Kerahasiaan informasi ini harus mutlak dijaga agar tidak sampai diketahui oleh orang lain.

Pada proses komunikasi atau pertukaran data, data akan melewati suatu media tertentu, baik media kabel, nirkabel atau keduanya. Tidak ada jaminan bahwa data tersebut tidak akan di sadap saat diperjalanan, sehingga kerahasiaan data akan terancam. Jika data-data tersebut merupakan data-data yang sangat rahasia dan sangat penting, dan data-data tersebut jatuh ke tangan rival atau saingan dari pemilik data, maka dengan mudah rival tersebut akan membahayakan pemilik data karena data rahasianya telah terbongkar.

Sebagai contoh, komunikasi rahasia seorang pejabat tentang penggelapan uang negara, dan KPK berhasil menyadap komunikasi tersebut. Maka KPK akan dengan mudah menangkap pejabat tersebut dengan barang bukti percakapan yang berhasil disadap. Maka pejabat tersebut akan mengalami kerugian yang sangat besar dalam hidupnya. Begitu juga jika data rahasia suatu perusahaan jatuh ke tangan perusahaan saingan, maka perusahaan saingan tersebut dapat membuat kebijakan-kebijakan yang sangat mengancam berdasarkan data rahasia yang berhasil didapatkan

### **3.3.2. Analisis Penyelesaian Masalah**

Salah satu penyelesaian terhadap masalah yang telah dianalisa, maka perlu adanya perlindungan terhadap data yang sedang dikirimkan, sehingga walaupun data berhasil jatuh ke tangan penyadap, tetapi data tersebut menjadi tidak berguna (kerahasiaan tetap terjaga).

Terdapat 3 cara yang dapat dilakukan, yaitu :

- 1) Disembunyikan pada suatu cover objek, sehingga data atau pesan rahasia tidak lagi tampak. Metode ini dikenal dengan nama steganografi.
- 2) Isi data/pesan diubah dan diacak sedemikian rupa dengan algoritma tertentu, sehingga tidak lagi memiliki arti. Pesan acak tersebut hanya bisa dikembalikan dengan suatu kunci tertentu. Mekanisme ini disebut dengan teknik kriptografi
- 3) Kombinasi dari steganografi dan kriptografi. Dimana data terlebih dahulu dienkripsi lalu kemudian disembunyikan ke dalam suatu file sebagai cover objek.

Pada penelitian ini, cara yang dipilih adalah kriptografi, yaitu dengan mengacak-acak (enkripsi) isi pesan hingga tidak memiliki arti dengan teknik kriptografi. Kriptografi adalah salah satu teknik untuk mengamankan data dengan cara mengubah bentuk data menjadi bentuk yang tidak dapat dikenali sehingga pihak asing hanya dapat mengetahui isi data tersebut jika memiliki kunci deskripsi untuk mengembalikan pesan tersebut menjadi pesan aslinya.

Kriptografi terbagi menjadi dua jenis, yaitu kriptografi kunci simetris dan kriptografi kunci asimetris. Kekurangan dari algoritma simetris adalah kunci yang digunakan untuk melakukan enkripsi dan deskripsi adalah sama, sehingga proses pertukaran kunci menjadi sangat sulit dilakukan dengan aman antara pengirim pesan dan penerima pesan. Sedangkan kriptografi kunci asimetris memiliki kunci enkripsi dan kunci deskripsi yang berbeda satu dengan yang lainnya.

Dengan menggunakan teknik kriptografi, maka secara otomatis proses pengiriman data akan bertambah sehingga akan menambah waktu pengiriman,

dimana data harus terlebih dahulu dienkripsi lalu kemudian dikirimkan, setelah diterima oleh penerima data, maka data tadi harus didekripsi terlebih dahulu, sehingga menambah waktu proses. Hal ini menjadi salah satu dampak negatif dari penerapan enkripsi.

Salah satu cara mengurangi dampak negatif kriptografi dalam waktu proses enkripsi dan dekripsi adalah dengan menggunakan algoritma kriptografi yang cepat, hemat, namun tetap memiliki keamanan yang tinggi. Salah satu teknik enkripsi dengan kriptografi yang paling sederhana, cepat, hemat, namun sangat kuat adalah dengan teknik XOR. Teknik XOR memiliki proses enkripsi dan dekripsi yang sangat sederhana namun sangat cepat dan hemat sumber daya komputer. Walau sederhana tetapi teknik XOR sangat kuat jika digunakan pada mode operasi yang tepat. Oleh sebab itu, penggunaan teknik XOR hanya menambah sedikit proses dan waktu.

Untuk meningkatkan keamanan dari *cipher text*, maka teknik XOR dapat dikombinasikan dengan mode operasi CBC (*Cipher Block Chaining*) dengan panjang blok sepanjang *256-bits*. Mode operasi CBC dipilih karena memiliki kelebihan pemetaan *cipher text* yang sulit ditebak, dimana karakter *cipher text* akan saling berkaitan dan tidak menghasilkan blok pemetaan yang sama dan sulit ditebak. Karena panjang blok yang dipilih untuk mode operasi CBC sepanjang *256-bits*, maka panjang kunci yang digunakan harus sepanjang *256-bits*. Untuk meningkatkan keamanan kunci maka kunci yang digunakan harus bersifat acak, sehingga kunci dibangkitkan secara acak.

Blok *256-bits* dipilih karena dengan panjang blok *256-bits* maka dibutuhkan kunci acak sepanjang *256-bits* juga. Sehingga kekuatan kunci sangat kuat, hal ini dikarenakan kombinasi kunci yang dapat terjadi sangat besar, yaitu terdapat  $2^{256}$  kemungkinan kunci yang dapat terjadi. Semua kombinasi kunci mungkin dapat digunakan karena kunci yang digunakan bersifat acak. Banyaknya kunci yang mungkin terjadi adalah:  $2^{256} = 1,16 \times 10^{77}$  kombinasi kunci yang mungkin terjadi.

Secara matematika, jika penyusup mampu melakukan operasi percobaan kunci dengan teknik brute force sebanyak  $10^{14}$  atau 100 triliun operasi percobaan perdetik, maka dibutuhkan waktu percobaan sebanyak:

$$\begin{aligned}\text{Waktu} &= 2^{256} / 10^{14} = 1,16 \times 10^{63} \text{ detik} \\ &= 3,67 \times 10^{55} \text{ Tahun}\end{aligned}$$

Berdasarkan hitungan matematika diatas maka dibutuhkan waktu triliunan tahun untuk menemukan kunci yang tepat. Sehingga berdasarkan analisis ini maka panjang blok *256-bits* telah cukup aman untuk diterapkan.

Untuk membangkitkan kunci secara acak, maka dibutuhkan algoritma pembangkit bilangan acak yang kuat sehingga menghasilkan deretan bilangan yang benar-benar acak sehingga kunci yang dihasilkan juga benar-benar acak. Salah satu algoritma pembangkit bilangan acak yang dapat digunakan adalah algoritma Blum Blum Shub. Algoritma ini mampu menghasilkan deretan bilangan acak yang sulit ditebak. Kunci yang dibangkitkan hanya digunakan untuk satu kali proses enkripsi dan dekripsi untuk meningkatkan keamanan pesan, sehingga jika ternyata kunci berhasil disadap atau dapatkan penyusup, maka



penyusup tidak dapat menggunakan kunci tersebut untuk mendekripsi *cipher text* lainnya.

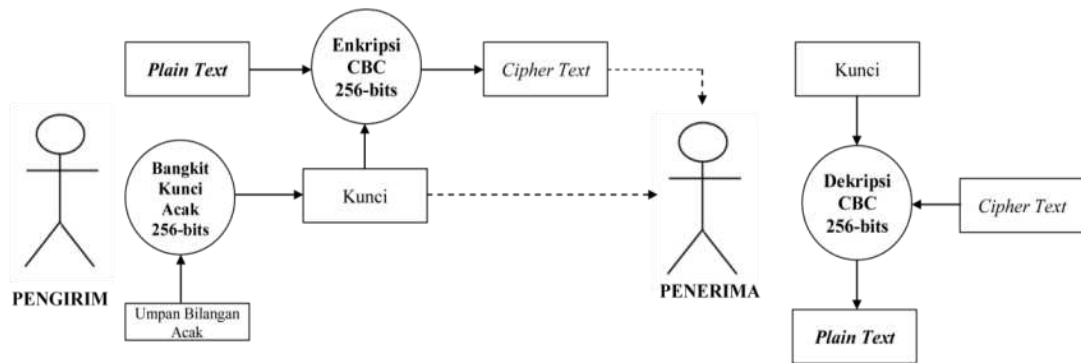
Kombinasi teknik XOR dengan mode operasi CBC (*Cipher Block Chaining*) 256-bits dan algoritma Blum Blum Shub diharapkan mampu menghasilkan algoritma kriptografi yang sangat kuat namun sangat cepat sehingga mampu mengamankan data tanpa banyak menambah waktu proses.

Sehingga berdasarkan analisa penyelesaian yang telah diuraikan, ada beberapa syarat yang harus dipenuhi :

- 1) Panjang Blok Cipher sepanjang 256-bits sehingga operasi dilakukan per 256-bits *plain text* atau *cipher text*.
- 2) Total panjang blok cipher mod 256 harus 0. Jika tidak, maka harus dilakukan *padding*, atau penambahan bit tambahan dengan bit '1' sampai  $plain\ text\ mod\ 256 = 0$ .
- 3) Panjang kunci harus sesuai dengan panjang blok, yaitu 256-bit dan harus benar-benar acak dimana kunci acak dibangkitkan dengan algoritma Blum Blum Shub. Kunci hanya boleh digunakan untuk sekali operasi enkripsi dan dekripsi untuk sebuah *plain text*, sedangkan untuk *plain text* yang lain harus menggunakan kunci yang lain.
- 4) Proses enkripsi dan dekripsi menggunakan teknik XOR agar proses menjadi cepat namun kuat

### 3.3.3. Analisis Proses

Analisis penyelesaian masalah yang telah dijelaskan dapat dirangkum ke dalam diagram proses berikut ini:



Gambar 3.1. Diagram Proses Penyelesaian Masalah yang Diusulkan  
Sumber: Oleh Penulis (2019)

Proses di atas terbagi menjadi dua buah area, yaitu area pengirim pesan dan area penerima pesan. Diagram proses di atas juga terbagi menjadi 3 buah tahap proses, yaitu tahap pertama untuk membangkitkan kunci acak 256-bits yang dilakukan pengirim data, tahap kedua pengirim data mengenkripsi *plain text* menjadi *cipher text* menggunakan kunci yang berhasil dibangkitkan, lalu kemudian kunci dan *cipher text* dikirim ke penerima, lalu tahap ketiga penerima data melakukan proses dekripsi terhadap *cipher text* yang diterima.

Pada penjelasan berikut ini, akan dilakukan proses perhitungan manual dari diagram proses di atas dimana *plain text* yang akan dienkripsi adalah:

*Plain text* = Gantungkanlah cita-citamu lebih tinggi dari langit.

Panjang = 51 karakter = 408 bits

### 1. Tahap Pembangkitan Kunci

Tahap pembangkitan kunci merupakan tahapan pembangkitan kunci acak 256-bits dengan menggunakan algoritma Blum Blum Shub. Langkah-langkah membangkitkan kunci adalah sebagai berikut:

- 1) Pilih bilangan prima  $p$  dan  $q$ . misal bilangan prima  $p$  dan  $q$  yang dipilih adalah:

$$p = 1987 \text{ dan } q = 1627.$$

$$2) n = p \cdot q = (1987)(1627) = 3232849$$

$$3) s = 2085153, \text{ karena } \text{GCD}(2085153, 3232849) = 1$$

Setelah itu, bangkitkan bit acak sebanyak 256 bit acak dengan persamaan  $x_i = x_{i-1}^2 \text{ mod } n$  lalu kemudian dimodulo dengan 2 untuk menghasilkan serangkaian bilangan acak 0 atau 1, dimana pada perhitungan pertama gunakan persamaan berikut:  $x_0 = s \text{ mod } p$

$$x_0 = 2085153^2 \text{ mod } 3232849 = 1180460 \text{ mod } 2 = 0$$

$$x_1 = 1180460^2 \text{ mod } 3232849 = 1811489 \text{ mod } 2 = 1$$

$$x_2 = 1811489^2 \text{ mod } 3232849 = 1951067 \text{ mod } 2 = 1$$

$$x_3 = 1951067^2 \text{ mod } 3232849 = 2138083 \text{ mod } 2 = 1$$

$$x_4 = 2138083^2 \text{ mod } 3232849 = 1717835 \text{ mod } 2 = 1$$

$$x_5 = 1717835^2 \text{ mod } 3232849 = 2821478 \text{ mod } 2 = 0$$

$$x_6 = 2821478^2 \text{ mod } 3232849 = 2618736 \text{ mod } 2 = 0$$

$$x_7 = 2618736^2 \text{ mod } 3232849 = 310976 \text{ mod } 2 = 0$$

$$x_8 = 310976^2 \text{ mod } 3232849 = 1860439 \text{ mod } 2 = 1$$

$$x_9 = 1860439^2 \text{ mod } 3232849 = 2887965 \text{ mod } 2 = 1$$

$$x_{10} = 2887965^2 \text{ mod } 3232849 = 1993048 \text{ mod } 2 = 0$$

$$x_{11} = 1993048^2 \text{ mod } 3232849 = 3202665 \text{ mod } 2 = 1$$

$$x_{12} = 3202665^2 \text{ mod } 3232849 = 2643287 \text{ mod } 2 = 1$$

$$x_{13} = 2643287^2 \text{ mod } 3232849 = 358760 \text{ mod } 2 = 0$$

$$\begin{aligned}
x_{14} &= 358760^2 \bmod 3232849 = 2553212 \bmod 2 = 0 \\
x_{15} &= 2553212^2 \bmod 3232849 = 219498 \bmod 2 = 0 \\
x_{16} &= 219498^2 \bmod 3232849 = 223357 \bmod 2 = 1 \\
x_{17} &= 223357^2 \bmod 3232849 = 2256530 \bmod 2 = 0 \\
x_{18} &= 2256530^2 \bmod 3232849 = 2960658 \bmod 2 = 0 \\
x_{19} &= 2960658^2 \bmod 3232849 = 739948 \bmod 2 = 0 \\
x_{20} &= 739948^2 \bmod 3232849 = 1270366 \bmod 2 = 0 \\
x_{21} &= 1270366^2 \bmod 3232849 = 1251703 \bmod 2 = 1 \\
x_{22} &= 1251703^2 \bmod 3232849 = 2159396 \bmod 2 = 0 \\
x_{23} &= 2159396^2 \bmod 3232849 = 809894 \bmod 2 = 0 \\
x_{24} &= 809894^2 \bmod 3232849 = 2626230 \bmod 2 = 0 \\
x_{25} &= 2626230^2 \bmod 3232849 = 1108038 \bmod 2 = 0 \\
x_{26} &= 1108038^2 \bmod 3232849 = 2679016 \bmod 2 = 0 \\
x_{27} &= 2679016^2 \bmod 3232849 = 1511618 \bmod 2 = 0 \\
x_{28} &= 1511618^2 \bmod 3232849 = 1606177 \bmod 2 = 1 \\
x_{29} &= 1606177^2 \bmod 3232849 = 751876 \bmod 2 = 0 \\
x_{30} &= 751876^2 \bmod 3232849 = 2146142 \bmod 2 = 0 \\
x_{31} &= 2146142^2 \bmod 3232849 = 1459790 \bmod 2 = 0 \\
x_{32} &= 1459790^2 \bmod 3232849 = 2700166 \bmod 2 = 0 \\
x_{33} &= 2700166^2 \bmod 3232849 = 788910 \bmod 2 = 0 \\
x_{34} &= 788910^2 \bmod 3232849 = 597167 \bmod 2 = 1 \\
x_{35} &= 597167^2 \bmod 3232849 = 2551246 \bmod 2 = 0 \\
x_{36} &= 2551246^2 \bmod 3232849 = 2851215 \bmod 2 = 1
\end{aligned}$$

$$\begin{aligned}
x_{37} &= 2851215^2 \bmod 3232849 = 1429657 \bmod 2 = 1 \\
x_{38} &= 1429657^2 \bmod 3232849 = 2082983 \bmod 2 = 1 \\
x_{39} &= 2082983^2 \bmod 3232849 = 1836842 \bmod 2 = 0 \\
x_{40} &= 1836842^2 \bmod 3232849 = 3044171 \bmod 2 = 1 \\
x_{41} &= 3044171^2 \bmod 3232849 = 2487345 \bmod 2 = 1 \\
x_{42} &= 2487345^2 \bmod 3232849 = 978181 \bmod 2 = 1 \\
x_{43} &= 978181^2 \bmod 3232849 = 2051684 \bmod 2 = 0 \\
x_{44} &= 2051684^2 \bmod 3232849 = 1839879 \bmod 2 = 1 \\
x_{45} &= 1839879^2 \bmod 3232849 = 2985402 \bmod 2 = 0 \\
x_{46} &= 2985402^2 \bmod 3232849 = 3090598 \bmod 2 = 0 \\
x_{47} &= 3090598^2 \bmod 3232849 = 945110 \bmod 2 = 0 \\
x_{48} &= 945110^2 \bmod 3232849 = 3199098 \bmod 2 = 0 \\
x_{49} &= 3199098^2 \bmod 3232849 = 1167153 \bmod 2 = 1 \\
x_{50} &= 1167153^2 \bmod 3232849 = 1145185 \bmod 2 = 1 \\
x_{51} &= 1145185^2 \bmod 3232849 = 1460338 \bmod 2 = 0 \\
x_{52} &= 1460338^2 \bmod 3232849 = 2670055 \bmod 2 = 1 \\
x_{53} &= 2670055^2 \bmod 3232849 = 1938510 \bmod 2 = 0 \\
x_{54} &= 1938510^2 \bmod 3232849 = 2602386 \bmod 2 = 0 \\
x_{55} &= 2602386^2 \bmod 3232849 = 1576970 \bmod 2 = 0 \\
x_{56} &= 1576970^2 \bmod 3232849 = 848989 \bmod 2 = 1 \\
x_{57} &= 848989^2 \bmod 3232849 = 2473326 \bmod 2 = 0 \\
x_{58} &= 2473326^2 \bmod 3232849 = 2379120 \bmod 2 = 0 \\
x_{59} &= 2379120^2 \bmod 3232849 = 932693 \bmod 2 = 1
\end{aligned}$$

$$\begin{aligned}
x_{60} &= 932693^2 \bmod 3232849 = 1826235 \bmod 2 = 1 \\
x_{61} &= 1826235^2 \bmod 3232849 = 1165714 \bmod 2 = 0 \\
x_{62} &= 1165714^2 \bmod 3232849 = 3079683 \bmod 2 = 1 \\
x_{63} &= 3079683^2 \bmod 3232849 = 2271212 \bmod 2 = 0 \\
x_{64} &= 2271212^2 \bmod 3232849 = 2194715 \bmod 2 = 1 \\
x_{65} &= 2194715^2 \bmod 3232849 = 262222 \bmod 2 = 0 \\
x_{66} &= 262222^2 \bmod 3232849 = 911903 \bmod 2 = 1 \\
x_{67} &= 911903^2 \bmod 3232849 = 730233 \bmod 2 = 1 \\
x_{68} &= 730233^2 \bmod 3232849 = 1188833 \bmod 2 = 1 \\
x_{69} &= 1188833^2 \bmod 3232849 = 3140314 \bmod 2 = 0 \\
x_{70} &= 3140314^2 \bmod 3232849 = 2142073 \bmod 2 = 1 \\
x_{71} &= 2142073^2 \bmod 3232849 = 399008 \bmod 2 = 0 \\
x_{72} &= 399008^2 \bmod 3232849 = 2502210 \bmod 2 = 0 \\
x_{73} &= 2502210^2 \bmod 3232849 = 2691498 \bmod 2 = 0 \\
x_{74} &= 2691498^2 \bmod 3232849 = 3143351 \bmod 2 = 1 \\
x_{75} &= 3143351^2 \bmod 3232849 = 2125031 \bmod 2 = 1 \\
x_{76} &= 2125031^2 \bmod 3232849 = 118046 \bmod 2 = 0 \\
x_{77} &= 118046^2 \bmod 3232849 = 1278926 \bmod 2 = 0 \\
x_{78} &= 1278926^2 \bmod 3232849 = 1460473 \bmod 2 = 1 \\
x_{79} &= 1460473^2 \bmod 3232849 = 2571962 \bmod 2 = 0 \\
x_{80} &= 2571962^2 \bmod 3232849 = 795473 \bmod 2 = 1 \\
x_{81} &= 795473^2 \bmod 3232849 = 2060412 \bmod 2 = 0 \\
x_{82} &= 2060412^2 \bmod 3232849 = 1124169 \bmod 2 = 1
\end{aligned}$$

$$\begin{aligned}
x_{83} &= 1124169^2 \bmod 3232849 = 2937971 \bmod 2 = 1 \\
x_{84} &= 2937971^2 \bmod 3232849 = 2328180 \bmod 2 = 0 \\
x_{85} &= 2328180^2 \bmod 3232849 = 1179570 \bmod 2 = 0 \\
x_{86} &= 1179570^2 \bmod 3232849 = 2736639 \bmod 2 = 1 \\
x_{87} &= 2736639^2 \bmod 3232849 = 885713 \bmod 2 = 1 \\
x_{88} &= 885713^2 \bmod 3232849 = 1147180 \bmod 2 = 0 \\
x_{89} &= 1147180^2 \bmod 3232849 = 247178 \bmod 2 = 0 \\
x_{90} &= 247178^2 \bmod 3232849 = 2583282 \bmod 2 = 0 \\
x_{91} &= 2583282^2 \bmod 3232849 = 2000254 \bmod 2 = 0 \\
x_{92} &= 2000254^2 \bmod 3232849 = 115079 \bmod 2 = 1 \\
x_{93} &= 115079^2 \bmod 3232849 = 1426737 \bmod 2 = 1 \\
x_{94} &= 1426737^2 \bmod 3232849 = 2162923 \bmod 2 = 1 \\
x_{95} &= 2162923^2 \bmod 3232849 = 2745972 \bmod 2 = 0 \\
x_{96} &= 2745972^2 \bmod 3232849 = 560204 \bmod 2 = 0 \\
x_{97} &= 560204^2 \bmod 3232849 = 2937790 \bmod 2 = 0 \\
x_{98} &= 2937790^2 \bmod 3232849 = 2422760 \bmod 2 = 0 \\
x_{99} &= 2422760^2 \bmod 3232849 = 1703713 \bmod 2 = 1 \\
x_{100} &= 1703713^2 \bmod 3232849 = 1881776 \bmod 2 = 0 \\
x_{101} &= 1881776^2 \bmod 3232849 = 2391969 \bmod 2 = 1 \\
x_{102} &= 2391969^2 \bmod 3232849 = 139667 \bmod 2 = 1 \\
x_{103} &= 139667^2 \bmod 3232849 = 3092872 \bmod 2 = 0 \\
x_{104} &= 3092872^2 \bmod 3232849 = 2495589 \bmod 2 = 1 \\
x_{105} &= 2495589^2 \bmod 3232849 = 473834 \bmod 2 = 0
\end{aligned}$$

$$\begin{aligned}
x_{106} &= 473834^2 \bmod 3232849 = 529355 \bmod 2 = 1 \\
x_{107} &= 529355^2 \bmod 3232849 = 3063252 \bmod 2 = 0 \\
x_{108} &= 3063252^2 \bmod 3232849 = 484856 \bmod 2 = 0 \\
x_{109} &= 484856^2 \bmod 3232849 = 2260003 \bmod 2 = 1 \\
x_{110} &= 2260003^2 \bmod 3232849 = 3096419 \bmod 2 = 1 \\
x_{111} &= 3096419^2 \bmod 3232849 = 1633207 \bmod 2 = 1 \\
x_{112} &= 1633207^2 \bmod 3232849 = 2819080 \bmod 2 = 0 \\
x_{113} &= 2819080^2 \bmod 3232849 = 2800868 \bmod 2 = 0 \\
x_{114} &= 2800868^2 \bmod 3232849 = 1074383 \bmod 2 = 1 \\
x_{115} &= 1074383^2 \bmod 3232849 = 396692 \bmod 2 = 0 \\
x_{116} &= 396692^2 \bmod 3232849 = 2384940 \bmod 2 = 0 \\
x_{117} &= 2384940^2 \bmod 3232849 = 2848869 \bmod 2 = 1 \\
x_{118} &= 2848869^2 \bmod 3232849 = 96057 \bmod 2 = 1 \\
x_{119} &= 96057^2 \bmod 3232849 = 396203 \bmod 2 = 1 \\
x_{120} &= 396203^2 \bmod 3232849 = 2601165 \bmod 2 = 1 \\
x_{121} &= 2601165^2 \bmod 3232849 = 589484 \bmod 2 = 0 \\
x_{122} &= 589484^2 \bmod 3232849 = 2145793 \bmod 2 = 1 \\
x_{123} &= 2145793^2 \bmod 3232849 = 383562 \bmod 2 = 0 \\
x_{124} &= 383562^2 \bmod 3232849 = 2548401 \bmod 2 = 1 \\
x_{125} &= 2548401^2 \bmod 3232849 = 148963 \bmod 2 = 1 \\
x_{126} &= 148963^2 \bmod 3232849 = 2932682 \bmod 2 = 0 \\
x_{127} &= 2932682^2 \bmod 3232849 = 726259 \bmod 2 = 1 \\
x_{128} &= 726259^2 \bmod 3232849 = 3122184 \bmod 2 = 0
\end{aligned}$$



$$\begin{aligned}
x_{129} &= 3122184^2 \bmod 3232849 = 710213 \bmod 2 = 1 \\
x_{130} &= 710213^2 \bmod 3232849 = 472993 \bmod 2 = 1 \\
x_{131} &= 472993^2 \bmod 3232849 = 2761551 \bmod 2 = 1 \\
x_{132} &= 2761551^2 \bmod 3232849 = 2448561 \bmod 2 = 1 \\
x_{133} &= 2448561^2 \bmod 3232849 = 3186261 \bmod 2 = 1 \\
x_{134} &= 3186261^2 \bmod 3232849 = 1200065 \bmod 2 = 1 \\
x_{135} &= 1200065^2 \bmod 3232849 = 2595950 \bmod 2 = 0 \\
x_{136} &= 2595950^2 \bmod 3232849 = 1840775 \bmod 2 = 1 \\
x_{137} &= 1840775^2 \bmod 3232849 = 112557 \bmod 2 = 1 \\
x_{138} &= 112557^2 \bmod 3232849 = 2775867 \bmod 2 = 1 \\
x_{139} &= 2775867^2 \bmod 3232849 = 201471 \bmod 2 = 1 \\
x_{140} &= 201471^2 \bmod 3232849 = 2144646 \bmod 2 = 0 \\
x_{141} &= 2144646^2 \bmod 3232849 = 2879056 \bmod 2 = 0 \\
x_{142} &= 2879056^2 \bmod 3232849 = 39267 \bmod 2 = 1 \\
x_{143} &= 39267^2 \bmod 3232849 = 3061165 \bmod 2 = 1 \\
x_{144} &= 3061165^2 \bmod 3232849 = 1511523 \bmod 2 = 1 \\
x_{145} &= 1511523^2 \bmod 3232849 = 2131343 \bmod 2 = 1 \\
x_{146} &= 2131343^2 \bmod 3232849 = 1375544 \bmod 2 = 0 \\
x_{147} &= 1375544^2 \bmod 3232849 = 2666065 \bmod 2 = 1 \\
x_{148} &= 2666065^2 \bmod 3232849 = 2363224 \bmod 2 = 0 \\
x_{149} &= 2363224^2 \bmod 3232849 = 205451 \bmod 2 = 1 \\
x_{150} &= 205451^2 \bmod 3232849 = 2036857 \bmod 2 = 1 \\
x_{151} &= 2036857^2 \bmod 3232849 = 194071 \bmod 2 = 1
\end{aligned}$$

$$\begin{aligned}
x_{152} &= 194071^2 \bmod 3232849 = 862191 \bmod 2 = 1 \\
x_{153} &= 862191^2 \bmod 3232849 = 2322874 \bmod 2 = 0 \\
x_{154} &= 2322874^2 \bmod 3232849 = 2256312 \bmod 2 = 0 \\
x_{155} &= 2256312^2 \bmod 3232849 = 1947198 \bmod 2 = 0 \\
x_{156} &= 1947198^2 \bmod 3232849 = 991383 \bmod 2 = 1 \\
x_{157} &= 991383^2 \bmod 3232849 = 2431105 \bmod 2 = 1 \\
x_{158} &= 2431105^2 \bmod 3232849 = 2842017 \bmod 2 = 1 \\
x_{159} &= 2842017^2 \bmod 3232849 = 769823 \bmod 2 = 1 \\
x_{160} &= 769823^2 \bmod 3232849 = 969743 \bmod 2 = 1 \\
x_{161} &= 969743^2 \bmod 3232849 = 1273288 \bmod 2 = 0 \\
x_{162} &= 1273288^2 \bmod 3232849 = 1488840 \bmod 2 = 0 \\
x_{163} &= 1488840^2 \bmod 3232849 = 2834562 \bmod 2 = 0 \\
x_{164} &= 2834562^2 \bmod 3232849 = 3099637 \bmod 2 = 1 \\
x_{165} &= 3099637^2 \bmod 3232849 = 328783 \bmod 2 = 1 \\
x_{166} &= 328783^2 \bmod 3232849 = 1489076 \bmod 2 = 0 \\
x_{167} &= 1489076^2 \bmod 3232849 = 861656 \bmod 2 = 0 \\
x_{168} &= 861656^2 \bmod 3232849 = 1426694 \bmod 2 = 0 \\
x_{169} &= 1426694^2 \bmod 3232849 = 2313652 \bmod 2 = 0 \\
x_{170} &= 2313652^2 \bmod 3232849 = 1874414 \bmod 2 = 0 \\
x_{171} &= 1874414^2 \bmod 3232849 = 3111535 \bmod 2 = 1 \\
x_{172} &= 3111535^2 \bmod 3232849 = 1157948 \bmod 2 = 0 \\
x_{173} &= 1157948^2 \bmod 3232849 = 50860 \bmod 2 = 0 \\
x_{174} &= 50860^2 \bmod 3232849 = 460400 \bmod 2 = 0
\end{aligned}$$

$$\begin{aligned}
x_{175} &= 460400^2 \bmod 3232849 = 3182466 \bmod 2 = 0 \\
x_{176} &= 3182466^2 \bmod 3232849 = 660224 \bmod 2 = 0 \\
x_{177} &= 660224^2 \bmod 3232849 = 1000959 \bmod 2 = 1 \\
x_{178} &= 1000959^2 \bmod 3232849 = 823299 \bmod 2 = 1 \\
x_{179} &= 823299^2 \bmod 3232849 = 2724967 \bmod 2 = 1 \\
x_{180} &= 2724967^2 \bmod 3232849 = 1569912 \bmod 2 = 0 \\
x_{181} &= 1569912^2 \bmod 3232849 = 3061312 \bmod 2 = 0 \\
x_{182} &= 3061312^2 \bmod 3232849 = 2783620 \bmod 2 = 0 \\
x_{183} &= 2783620^2 \bmod 3232849 = 2561314 \bmod 2 = 0 \\
x_{184} &= 2561314^2 \bmod 3232849 = 2683517 \bmod 2 = 1 \\
x_{185} &= 2683517^2 \bmod 3232849 = 1822017 \bmod 2 = 1 \\
x_{186} &= 1822017^2 \bmod 3232849 = 1200018 \bmod 2 = 0 \\
x_{187} &= 1200018^2 \bmod 3232849 = 2941764 \bmod 2 = 0 \\
x_{188} &= 2941764^2 \bmod 3232849 = 737784 \bmod 2 = 0 \\
x_{189} &= 737784^2 \bmod 3232849 = 745979 \bmod 2 = 1 \\
x_{190} &= 745979^2 \bmod 3232849 = 1438675 \bmod 2 = 1 \\
x_{191} &= 1438675^2 \bmod 3232849 = 2676110 \bmod 2 = 0 \\
x_{192} &= 2676110^2 \bmod 3232849 = 2450548 \bmod 2 = 0 \\
x_{193} &= 2450548^2 \bmod 3232849 = 374656 \bmod 2 = 0 \\
x_{194} &= 374656^2 \bmod 3232849 = 47605 \bmod 2 = 1 \\
x_{195} &= 47605^2 \bmod 3232849 = 8876 \bmod 2 = 0 \\
x_{196} &= 8876^2 \bmod 3232849 = 1195000 \bmod 2 = 0 \\
x_{197} &= 1195000^2 \bmod 3232849 = 1241173 \bmod 2 = 1
\end{aligned}$$

$$\begin{aligned}
x_{198} &= 1241173^2 \bmod 3232849 = 2908996 \bmod 2 = 0 \\
x_{199} &= 2908996^2 \bmod 3232849 = 678351 \bmod 2 = 1 \\
x_{200} &= 678351^2 \bmod 3232849 = 2818239 \bmod 2 = 1 \\
x_{201} &= 2818239^2 \bmod 3232849 = 1172223 \bmod 2 = 1 \\
x_{202} &= 1172223^2 \bmod 3232849 = 458524 \bmod 2 = 0 \\
x_{203} &= 458524^2 \bmod 3232849 = 2389559 \bmod 2 = 1 \\
x_{204} &= 2389559^2 \bmod 3232849 = 1763872 \bmod 2 = 0 \\
x_{205} &= 1763872^2 \bmod 3232849 = 2280368 \bmod 2 = 0 \\
x_{206} &= 2280368^2 \bmod 3232849 = 1804736 \bmod 2 = 0 \\
x_{207} &= 1804736^2 \bmod 3232849 = 2524988 \bmod 2 = 0 \\
x_{208} &= 2524988^2 \bmod 3232849 = 1463113 \bmod 2 = 1 \\
x_{209} &= 1463113^2 \bmod 3232849 = 795590 \bmod 2 = 0 \\
x_{210} &= 795590^2 \bmod 3232849 = 709541 \bmod 2 = 1 \\
x_{211} &= 709541^2 \bmod 3232849 = 88760 \bmod 2 = 0 \\
x_{212} &= 88760^2 \bmod 3232849 = 3117436 \bmod 2 = 0 \\
x_{213} &= 3117436^2 \bmod 3232849 = 822689 \bmod 2 = 1 \\
x_{214} &= 822689^2 \bmod 3232849 = 855477 \bmod 2 = 1 \\
x_{215} &= 855477^2 \bmod 3232849 = 1472305 \bmod 2 = 1 \\
x_{216} &= 1472305^2 \bmod 3232849 = 1800092 \bmod 2 = 0 \\
x_{217} &= 1800092^2 \bmod 3232849 = 1395878 \bmod 2 = 0 \\
x_{218} &= 1395878^2 \bmod 3232849 = 1737245 \bmod 2 = 1 \\
x_{219} &= 1737245^2 \bmod 3232849 = 471773 \bmod 2 = 1 \\
x_{220} &= 471773^2 \bmod 3232849 = 1041275 \bmod 2 = 1
\end{aligned}$$

$$\begin{aligned}
x_{221} &= 1041275^2 \bmod 3232849 = 1330911 \bmod 2 = 1 \\
x_{222} &= 1330911^2 \bmod 3232849 = 862935 \bmod 2 = 1 \\
x_{223} &= 862935^2 \bmod 3232849 = 2375565 \bmod 2 = 1 \\
x_{224} &= 2375565^2 \bmod 3232849 = 2594939 \bmod 2 = 1 \\
x_{225} &= 2594939^2 \bmod 3232849 = 765923 \bmod 2 = 1 \\
x_{226} &= 765923^2 \bmod 3232849 = 2029540 \bmod 2 = 0 \\
x_{227} &= 2029540^2 \bmod 3232849 = 1509418 \bmod 2 = 0 \\
x_{228} &= 1509418^2 \bmod 3232849 = 2064521 \bmod 2 = 1 \\
x_{229} &= 2064521^2 \bmod 3232849 = 646559 \bmod 2 = 1 \\
x_{230} &= 646559^2 \bmod 3232849 = 2069140 \bmod 2 = 0 \\
x_{231} &= 2069140^2 \bmod 3232849 = 820524 \bmod 2 = 0 \\
x_{232} &= 820524^2 \bmod 3232849 = 2666081 \bmod 2 = 1 \\
x_{233} &= 2666081^2 \bmod 3232849 = 390637 \bmod 2 = 1 \\
x_{234} &= 390637^2 \bmod 3232849 = 327271 \bmod 2 = 1 \\
x_{235} &= 327271^2 \bmod 3232849 = 2020071 \bmod 2 = 1 \\
x_{236} &= 2020071^2 \bmod 3232849 = 564848 \bmod 2 = 0 \\
x_{237} &= 564848^2 \bmod 3232849 = 162445 \bmod 2 = 1 \\
x_{238} &= 162445^2 \bmod 3232849 = 1864487 \bmod 2 = 1 \\
x_{239} &= 1864487^2 \bmod 3232849 = 147828 \bmod 2 = 0 \\
x_{240} &= 147828^2 \bmod 3232849 = 2291193 \bmod 2 = 1 \\
x_{241} &= 2291193^2 \bmod 3232849 = 500069 \bmod 2 = 1 \\
x_{242} &= 500069^2 \bmod 3232849 = 1668913 \bmod 2 = 1 \\
x_{243} &= 1668913^2 \bmod 3232849 = 3079921 \bmod 2 = 1
\end{aligned}$$

$$\begin{aligned}
x_{244} &= 3079921^2 \bmod 3232849 = 543518 \bmod 2 = 0 \\
x_{245} &= 543518^2 \bmod 3232849 = 540402 \bmod 2 = 0 \\
x_{246} &= 540402^2 \bmod 3232849 = 1372887 \bmod 2 = 1 \\
x_{247} &= 1372887^2 \bmod 3232849 = 3090789 \bmod 2 = 1 \\
x_{248} &= 3090789^2 \bmod 3232849 = 1600142 \bmod 2 = 0 \\
x_{249} &= 1600142^2 \bmod 3232849 = 2450825 \bmod 2 = 1 \\
x_{250} &= 2450825^2 \bmod 3232849 = 258397 \bmod 2 = 1 \\
x_{251} &= 258397^2 \bmod 3232849 = 979212 \bmod 2 = 0 \\
x_{252} &= 979212^2 \bmod 3232849 = 2826091 \bmod 2 = 1 \\
x_{253} &= 2826091^2 \bmod 3232849 = 1324442 \bmod 2 = 0 \\
x_{254} &= 1324442^2 \bmod 3232849 = 2743964 \bmod 2 = 0 \\
x_{255} &= 2743964^2 \bmod 3232849 = 783806 \bmod 2 = 0
\end{aligned}$$

Gabungkan seluruh bit acak yang diperoleh untuk mendapatkan kunci acak 256-bits sebagai berikut:

Kunci :

```

011110001101100010000100000010000010111011101000011010001001
101010111010001100101011001100001110000101101010011100100111
101011010111111011110011110101111000111110001100000100000111
000011000110001001011101000010100111001111111100110011110110
1111001101101000

```

Lalu konversi kunci tersebut ke dalam bentuk bilangan heksadesimal sebagai berikut:

Kunci :

78d884082ee8689aba32b30e16a727ad7ef3d78f8c1070c625d0a73fccf6f36

8

## 2. Tahap Enkripsi

Tahap enkripsi merupakan tahapan untuk melakukan proses enkripsi terhadap *plain text* dengan teknik XOR pada mode operasi CBC 256-bits..

Misal, terdapat sebuah pesan atau *plain text* yang sangat rahasia yang akan dikirimkan, maka pesan ini harus dienkripsi terlebih dahulu agar terjaga kerahasiaannya. Andai pesan tersebut adalah:

*Plain text* = Gantungkanlah cita-citamu lebih tinggi dari langit.

Pesan tersebut memiliki:

Panjang Karakter = 51

Panjang Biner = 408

Kemudian konversi setiap karakter dari *plain text* ke dalam bentuk biner 8-bits, sehingga menjadi:

G = 01000111

a = 01100001

n = 01101110

t = 01110100

u = 01110101

n = 01101110

g = 01100111

k = 01101011

a = 01100001  
n = 01101110  
l = 01101100  
a = 01100001  
h = 01101000  
spasi = 00100000  
c = 01100011  
i = 01101001  
t = 01110100  
a = 01100001  
- = 00101101  
c = 01100011  
i = 01101001  
t = 01110100  
a = 01100001  
m = 01101101  
u = 01110101  
spasi = 00100000  
l = 01101100  
e = 01100101  
b = 01100010  
i = 01101001  
h = 01101000



spasi = 00100000

t = 01110100

i = 01101001

n = 01101110

g = 01100111

g = 01100111

i = 01101001

spasi = 00100000

d = 01100100

a = 01100001

r = 01110010

i = 01101001

spasi = 00100000

l = 01101100

a = 01100001

n = 01101110

g = 01100111

i = 01101001

t = 01110100

. = 00101110

Sehingga *plain text* dalam bentuk biner menjadi:

*Plain text* =

```
010001110110000101101110011101000111010101101110011001110110
101101100001011011100110110001100001011010000010000001100011
011010010111010001100001001011010110001101101001011101000110
000101101101011101010010000001101100011001010110001001101001
011010000010000001110100011010010110111001100111011001110110
100100100000011001000110000101110010011010010010000001101100
011000010110111001100111011010010111010000101110
```

Mengingat mode operasi blok cipher CBC yang dipilih beroperasi dengan panjang blok *256-bits*, maka *plain text* dibagi ke dalam blok-blok berukuran *256-bits*, dimana jika blok terakhir panjangnya kurang dari *256-bits*, maka ditambahkan bit padding '1' sepanjang yang dibutuhkan sehingga panjang blok terakhir tetap *256-bits*. Untuk menghitung jumlah bit padding maka dapat dilakukan perhitungan sebagai berikut:

$$\text{Bit padding} = 256 - (\text{panjang biner } \textit{plain text} \text{ mod } 256)$$

$$\text{Bit padding} = 256 - (408 \text{ mod } 256)$$

$$\text{Bit padding} = 256 - 152$$

$$\text{Bit padding} = 104$$

Sehingga berdasarkan perhitungan di atas, maka *plain text* perlu ditambahkan bit padding "1" sebanyak 104 buah bit.

Seperti contoh berikut ini:



101011010111111011110011110101111000111110001100000100000111  
 000011000110001001011101000010100111001111111100110011110110  
 1111001101101000

Tentukan nilai awal dari *Initial Vector* (IV), yaitu deretan bit “0” sepanjang 256-bits.

*Initial Vector* (IV) =

00  
 00  
 00  
 00  
 0000000000000000

Lakukan proses XOR terhadap blok ke - 0 dari biner *plain text* dengan IV awal. Proses tersebut dapat dilihat pada tabel berikut:

Tabel 3.1. Proses XOR Plain text Blok ke - 0 dengan Initial Vector (IV)

<b>Blok Pesan</b>	<i>Plain text</i>	<i>Initial Vector</i> (IV)	<i>Hasil IV</i>
Blok ke - 0	010001110110000101	000000000000000000	010001110110000101
	101110011101000111	000000000000000000	101110011101000111
	010101101110011001	000000000000000000	010101101110011001
	110110101101100001	000000000000000000	110110101101100001
	011011100110110001	000000000000000000	011011100110110001
	100001011010000010	000000000000000000	100001011010000010
	000001100011011010	000000000000000000	000001100011011010
	010111010001100001	000000000000000000	010111010001100001
	001011010110001101	000000000000000000	001011010110001101
	101001011101000110	000000000000000000	101001011101000110
	000101101101011101	000000000000000000	000101101101011101
	010010000001101100	000000000000000000	010010000001101100
	011001010110001001	000000000000000000	011001010110001001
	101001011010000010	000000000000000000	101001011010000010
	0000	0000	0000

Sumber: Oleh Penulis (2019)

Proses di atas menghasilkan deretan hasil operasi XOR dari *plain text* dengan IV yang menghasilkan hasil IV. Kemudian hasil IV dilakukan proses XOR kembali dengan kunci acak 256-bits. Proses tersebut dapat dilihat pada tabel berikut:

Tabel 3.2. Proses XOR Plain text Blok ke - 0 dengan Kunci Acak

Blok Pesan	Hasil IV	Kunci	Cipher text
Blok ke - 0	010001110110000101 101110011101000111 010101101110011001 110110101101100001 011011100110110001 100001011010000010 000001100011011010 010111010001100001 001011010110001101 101001011101000110 000101101101011101 010010000001101100 011001010110001001 101001011010000010 0000	011110001101100010 000100000010000010 111011101000011010 001001101010111010 001100101011001100 001110000101101010 011100100111101011 010111111011110011 110101111000111110 001100000100000111 000011000110001001 011101000010100111 001111111100110011 110110111100110110 1000	001111111011100111 101010011111000101 101110000110000011 111111000111011011 010111001101111101 101111011111101000 011101000100110001 000000101010010010 111110101110110011 100101011001000001 000110101011010100 001111000011001011 010110101010111010 011111100110110100 1000

Sumber: Oleh Penulis (2019)

Hasil operasi di atas menghasilkan *cipher text* blok ke – 0 yaitu:

```
0011111110111001111010100111110001011011100001100000111111111
000111011011010111001101111101101111011111101000011101000100
110001000000101010010010111110101110110011100101011001000001
0001101010110101000011110000110010110101101010101110100111111
1001101101001000
```

Kemudian inisialisasikan nilai dari *Initial Vector* (IV) yang baru, dimana *cipher text* blok ke – 0 atau *cipher text* sebelumnya dijadikan sebagai *Initial Vector* (IV) yang baru untuk proses blok selanjutnya, sehingga:

*Initial Vector (IV) =*

```
0011111110111001111010100111110001011011100001100000111111111
00011101101101011100110111110110111101111101000011101000100
110001000000101010010010111110101110110011100101011001000001
000110101011010100001111000011001011010110101010111010011111
1001101101001000
```

Lakukan proses di atas untuk blok selanjutnya, yaitu blok ke – 1. Dimana *plain text* dari blok ke – 1 di lakukan operasi XOR dengan *Initial Vector (IV)* terbaru. Proses tersebut dapat dilihat pada tabel berikut:

*Tabel 3.3. Proses XOR Plain text Blok ke - 1 dengan Initial Vector (IV)*

<b>Blok Pesan</b>	<i>Plain text</i>	<i>Initial Vector (IV)</i>	<i>Hasil IV</i>
Blok ke - 1	011101000110100101	001111111011100111	010010111101000010
	101110011001110110	101010011111000101	000100000110110011
	011101101001001000	101110000110000011	110011101111001011
	000110010001100001	111111000111011011	111001010110111010
	011100100110100100	010111001101111101	001011101011011001
	100000011011000110	101111011111101000	001111000100101110
	000101101110011001	011101000100110001	011000101010101000
	110110100101110100	000000101010010010	110110001111100110
	001011101111111111	111110101110110011	110101000001001100
	111111111111111111	100101011001000001	011010100110111110
	111111111111111111	000110101011010100	111001010100101011
	111111111111111111	001111000011001011	110000111100110100
	111111111111111111	010110101010111010	101001010101000101
	111111111111111111	011111100110110100	100000011001001011
	1111	1000	0111

*Sumber: Oleh Penulis (2019)*

Proses di atas menghasilkan deretan hasil operasi XOR dari *plain text* dengan IV yang menghasilkan hasil IV. Kemudian hasil IV dilakukan proses XOR kembali dengan kunci acak 256-bits. Proses tersebut dapat dilihat pada tabel berikut:

Tabel 3.4. Proses XOR Plain text Blok ke - 1 dengan Kunci Acak

Blok Pesan	Hasil IV	Kunci	Cipher text
Blok ke - 1	010010111101000010 000100000110110011 110011101111001011 111001010110111010 001011101011011001 001111000100101110 011000101010101000 110110001111100110 110101000001001100 011010100110111110 111001010100101011 110000111100110100 101001010101000101 100000011001001011 0111	011110001101100010 000100000010000010 111011101000011010 001001101010111010 001100101011001100 001110000101101010 011100100111101011 010111111011110011 110101111000111110 001100000100000111 000011000110001001 011101000010100111 001111111100110011 110110111100110110 1000	001100110000100000 000000000100110001 001000000111010001 110000111100000000 000111000000010101 001100101011010001 100100101101000101 10001110100010101 00000111001110010 010110100010111001 111010010010100010 101101111110010011 100110101001110110 010110100101111101 1111

Sumber: Oleh Penulis (2019)

Hasil operasi di atas menghasilkan *cipher text* blok ke – 1 yaitu:

```
00110011000010000000000000000000100110001001000000111010001110000
111100000000000111000000010101000001000001000100000100001101
000011100001110100010101000000111001110010010110100010111001
111010010010100010101101111110010011100110101001110110010110
1001011111011111
```

Karena blok *plain text* telah habis, maka proses dihentikan, kemudian gabungkan seluruh *cipher text* yang diperoleh dari setiap blok untuk membentuk *cipher text* yang utuh sebagai berikut:

*Cipher text* =

```
0011111110111001111010100111110001011011100001100000111111111
000111011011010111001101111101101110111111101000011101000100
110001000000101010010010111110101110110011100101011001000001
```

```

000110101011010100001111000011001011010110101010111010011111
10011011010010000011001100001000000000000000000010011000100100000
011101000111000011110000000000011100000001010100000100000100
010000010000110100001110000111010001010100000011100111001001
011010001011100111101001001010001010110111111001001110011010
10011101100101101001011111011111

```

Lalu konversikan seluruh bit dari *cipher text* ke dalam bentuk heksadesimal, sehingga menjadi:

*Cipher text* =

```

3fb9ea7c5b860ff1db5cdf6f7e8744c40a92faece56411ab50f0cb5aae9f9b48
330800131207470f001c054104410d0e1d15039c968b9e928adf939a9d969
7df

```

### 3. Tahap Dekripsi

Proses dekripsi dilakukan oleh penerima pesan setelah penerima pesan menerima dua buah data dari pengirim pesan, yaitu *cipher text* dan kunci. Tahap proses dekripsi dapat dijelaskan sebagai berikut:

Konversi *cipher text* yang diterima oleh penerima data dari bentuk heksadesimal ke dalam bentuk biner.

*Cipher text* Heksadesimal =

```

3fb9ea7c5b860ff1db5cdf6f7e8744c40a92faece56411ab50f0cb5aae9f9b48
330800131207470f001c054104410d0e1d15039c968b9e928adf939a9d9697df

```





*Cipher text* Blok ke – 0 :

```
0011111110111001111010100111110001011011100001100000111111111
00011101101101011100110111110110111101111110100001110100010011
00010000001010100100101111101011101100111001010110010000010001
10101011010100001111000011001011010110101010111010011111100110
1101001000
```

*Cipher text* Blok ke – 1 :

```
0011001100001000000000000000100110001001000000111010001110000
11110000000000011100000001010100000100000100010000010000110100
00111000011101000101010000001110011100100101101000101110011110
1001001010001010110111110010011100110101001110110010110100101
1111011111
```

Kemudian lakukan operasi XOR dari blok ke – 0 dengan kunci biner, kemudian hasilnya di proses XOR kembali dengan *Initial Vector* (IV) pertama yaitu serangkaian bit 0 sepanjang 256-bits.

*Initial Vector* (IV) pertama =

```
0000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000
0000000000
```

Adapun proses XOR *cipher text* blok ke – 0 dapat dilihat pada tabel berikut:

Tabel 3.5. Proses XOR Cipher Text Blok ke - 0 dengan Kunci Acak

<b>Blok Cipher Text</b>	<b>Cipher Text</b>	<b>Kunci</b>	<b>Hasil XOR</b>
Blok ke - 0	001111111011100111 101010011111000101 101110000110000011 111111000111011011 010111001101111101 101111011111101000 011101000100110001 000000101010010010 111110101110110011 100101011001000001 000110101011010100 001111000011001011 010110101010111010 011111100110110100 1000	011110001101100010 00010000010000010 111011101000011010 001001101010111010 001100101011001100 001110000101101010 011100100111101011 010111111011110011 110101111000111110 001100000100000111 000011000110001001 011101000010100111 001111111100110011 110110111100110110 1000	010001110110000101 101110011101000111 010101101110011001 110110101101100001 011011100110110001 100001011010000010 000001100011011010 010111010001100001 001011010110001101 101001011101000110 000101101101011101 010010000001101100 011001010110001001 101001011010000010 0000

Sumber: Oleh Penulis (2019)

Kemudian hasil XOR antara *cipher text* blok ke – 0 dengan kunci di-XOR-kan kembali dengan *Initial Vector* (IV) pertama untuk mendapatkan *plain text* seperti yang ditunjukkan pada tabel berikut:

Tabel 3.6. Proses XOR Hasil Blok ke – 0 dengan Initial Vector (IV)

<b>Blok Cipher Text</b>	<b>Hasil XOR</b>	<b>Initial Vector (IV)</b>	<b>Plain Text</b>
Blok ke - 0	001111111011100111 101010011111000101 101110000110000011 111111000111011011 010111001101111101 101111011111101000 011101000100110001 000000101010010010 111110101110110011 100101011001000001 000110101011010100 001111000011001011 010110101010111010 011111100110110100 1000	000000000000000000 000000000000000000 000000000000000000 000000000000000000 000000000000000000 000000000000000000 000000000000000000 000000000000000000 000000000000000000 000000000000000000 000000000000000000 000000000000000000 000000000000000000 000000000000000000 0000	010001110110000101 101110011101000111 010101101110011001 110110101101100001 011011100110110001 100001011010000010 000001100011011010 010111010001100001 001011010110001101 101001011101000110 000101101101011101 010010000001101100 011001010110001001 101001011010000010 0000

Sumber: Oleh Penulis (2019)

Selanjutnya nilai dari *Initial Vector* (IV) berubah digantikan dengan nilai dari hasil XOR antara *cipher text* blok sebelumnya (blok ke – 0) dengan kunci sehingga *Initial Vector* (IV) berubah menjadi:

*Initial Vector* (IV) kedua =

```
0011111110111001111010100111110001011011100001100000111111111
00011101101101011100110111110110111101111110100001110100010011
00010000001010100100101111101011101100111001010110010000010001
10101011010100001111000011001011010110101010111010011111100110
1101001000
```

*Plain text* dari blok ke – 0 telah didapatkan yaitu:

*Plain text* Blok Ke – 0 =

```
010001110110000101101110011101000111010101101110011001110110
10110110000101101110011011000110000101101000001000000110001101
10100101110100011000010010110101100011011010010111010001100001
01101101011101010010000001101100011001010110001001101001011010
0000100000
```

Selanjutnya lakukan proses yang sama untuk *cipher text* blok selanjutnya, dalam hal ini adalah *cipher text* blok ke – 1. Proses dekripsi *cipher text* blok ke – 1 mula-mula dilakukan proses XOR dengan kunci seperti pada tabel berikut:

Tabel 3.7. Proses XOR Cipher Text Blok ke - 1 dengan Kunci Acak

<b>Blok Cipher Text</b>	<b>Cipher text</b>	<b>Kunci</b>	<b>Hasil XOR</b>
Blok ke - 1	001100110000100000 000000000100110001 001000000111010001 110000111100000000 000111000000010101 000001000001000100 000100001101000011 100001110100010101 000000111001110010 010110100010111001 111010010010100010 101101111110010011 100110101001110110 010110100101111101 1111	011110001101100010 00010000010000010 111011101000011010 001001101010111010 001100101011001100 001110000101101010 011100100111101011 010111111011110011 110101111000111110 001100000100000111 000011000110001001 011101000010100111 001111111100110011 110110111100110110 110110111100110111	0100101111101000010 000100000110110011 110011101111001011 111001010110111010 001011101011011001 001111000100101110 011000101010101000 110110001111100110 110101000001001100 011010100110111110 111001010100101011 110000111100110100 101001010101000101 100000011001001011 0111

Sumber: Oleh Penulis (2019)

Kemudian hasil XOR antara *cipher text* blok ke – 1 dengan kunci di-XOR-kan kembali dengan *Initial Vector* (IV) kedua untuk mendapatkan *plain text* seperti yang ditunjukkan pada tabel berikut:

Tabel 3.8. Proses XOR Blok ke – 1 Hasil dengan Initial Vector (IV)

<b>Blok Cipher Text</b>	<b>Hasil XOR</b>	<b>Initial Vector (IV)</b>	<b>Plain text</b>
Blok ke - 1	0100101111101000010 000100000110110011 110011101111001011 111001010110111010 001011101011011001 001111000100101110 011000101010101000 110110001111100110 110101000001001100 011010100110111110 111001010100101011 110000111100110100 101001010101000101 100000011001001011 0111	001111111011100111 101010011111000101 101110000110000011 111111000111011011 010111001101111101 101111011111101000 011101000100110001 000000101010010010 111110101110110011 100101011001000001 000110101011010100 001111000011001011 010110101010111010 011111100110110100 1000	011101000110100101 101110011001110110 011101101001001000 000110010001100001 011100100110100100 100000011011000110 000101101110011001 110110100101110100 001011101111111111 111111111111111111 111111111111111111 111111111111111111 111111111111111111 111111111111111111 111111111111111111 1111

Sumber: Oleh Penulis (2019)



Kemudian bagi *plain text* ke dalam masing-masing 8 *bits* kemudian konversi setiap 8 *bits* tersebut ke dalam karakter ASCII, namun dengan catatan jika 8 *bits* tersebut seluruhnya terdiri dari bit 1 maka diabaikan. Proses konversi setiap 8 *bits* ke karakter ASCII dapat dilihat sebagai berikut:

01000111 = 71 = G

01100001 = 97 = a

01101110 = 110 = n

01110100 = 116 = t

01110101 = 117 = u

01101110 = 110 = n

01100111 = 103 = g

01101011 = 107 = k

01100001 = 97 = a

01101110 = 110 = n

01101100 = 108 = l

01100001 = 97 = a

01101000 = 104 = h

00100000 = 32 = spasi

01100011 = 99 = c

01101001 = 105 = i

01110100 = 116 = t

01100001 = 97 = a

00101101 = 45 = -

01100011 = 99 = c  
01101001 = 105 = i  
01110100 = 116 = t  
01100001 = 97 = a  
01101101 = 109 = m  
01110101 = 117 = u  
00100000 = 32 = spasi  
01101100 = 108 = l  
01100101 = 101 = e  
01100010 = 98 = b  
01101001 = 105 = i  
01101000 = 104 = h  
00100000 = 32 = spasi  
01110100 = 116 = t  
01101001 = 105 = i  
01101110 = 110 = n  
01100111 = 103 = g  
01100111 = 103 = g  
01101001 = 105 = i  
00100000 = 32 = spasi  
01100100 = 100 = d  
01100001 = 97 = a  
01110010 = 114 = r





Kemudian gabungkan seluruh karakter ASCII yang berhasil ditemukan untuk mendapatkan *plain text* dalam bentuk karakter sebagai berikut:

*Plain text* = Gantungkanlah cita-citamu lebih tinggi dari langit.

### 3.3.4. Analisis Kebutuhan Sistem

Beberapa hal yang dibutuhkan untuk menjalankan sistem ini adalah :

#### 1. Hardware

##### a. Seperangkat Komputer

Seperangkat komputer yang meliputi : processor, RAM, harddisk, monitor, *mouse*, dan *keyboard* yang digunakan sebagai sistem komputer oleh aplikasi saat dijalankan dan juga digunakan untuk membangun aplikasi. Komputer digunakan sebagai media pengujian algoritma yang dirancang.

#### 2. Software

##### a. Sistem Operasi

Sistem operasi dibutuhkan sebagai *software* induk utama pada komputer. Sistem operasi yang dapat digunakan adalah Windows XP, 7, 8, dan 10. Atau juga bisa menggunakan Linux dengan berbagai varian distribusinya. Sistem operasi menjadi dasar dimana aplikasi yang dirancang dijalankan.

##### b. XAMPP

Sebuah aplikasi paket yang di dalamnya terdapat software PHP dan DBMS MySQL. PHP digunakan untuk membangun aplikasi yang berbasis *web*.

c. *Text Editor*

*Text editor* digunakan untuk mengetikkan *source code* dari aplikasi atau pengkodean dari aplikasi yang akan dibangun.

d. *Browser*

*Browser* digunakan untuk menjalankan aplikasi yang berbasis *web*. Dikarenakan aplikasi yang akan dibangun merupakan jenis aplikasi berbasis *web*, maka sebuah *browser* dibutuhkan untuk menjalankan aplikasi tersebut. *Browser* yang dapat digunakan adalah Mozilla Firefox, Google Chrome, Opera Mini, dan berbagai jenis *browser* lainnya.

### 3.4. Rancangan Penelitian

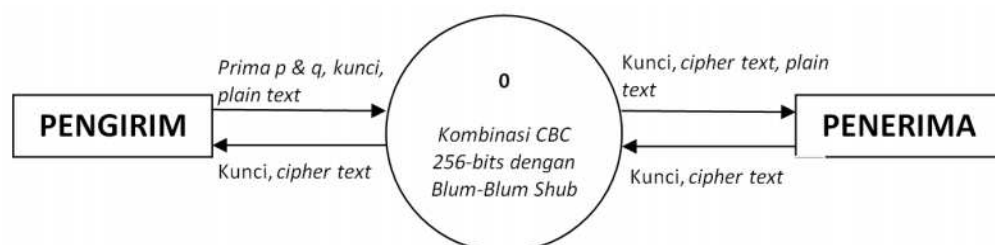
Rancangan penelitian dilakukan terhadap 3 objek, yaitu rancangan sistem, rancangan database, rancangan *interface*, dan *flowchart* proses.

#### 3.4.1. Rancangan Sistem

Pemodelan sistem yang akan diberikan dalam bentuk diagram konteks dan DFD (*Data Flow Diagram*).

##### 1. Diagram Konteks

Berikut merupakan diagram konteks dari sistem yang akan dibangun:



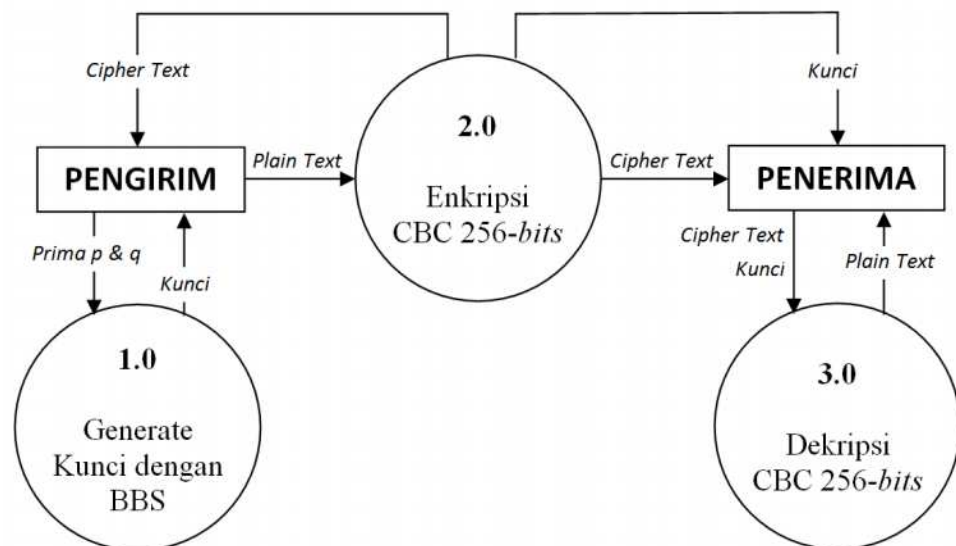
Gambar 3.2. Diagram Konteks dari Sistem

Sumber: Oleh Penulis (2019)

Pada diagram di atas, terdapat aliran data dari pengirim ke sistem dan dari penerima ke sistem. Pengirim dapat mengalirkan data bilangan prima  $p$  dan  $q$ , kunci dan *plain text*, kemudian sistem dapat mengalirkan data kunci, dan *cipher text* ke pengirim. Penerima juga dapat mengalirkan data kunci dan *cipher text*, kemudian sistem dapat mengalirkan data kunci, *plain text* dan *cipher text*.

## 2. Data Flow Diagram

Berikut merupakan DFD Level 1 dari sistem yang dirancang:



Gambar 3.3. DFD Level 1 dari Sistem

Sumber: Oleh Penulis (2019)

Pada gambar diagram DFD diatas merupakan penjabaran dari diagram konteks dari sistem agar alur data dan proses yang ada menjadi lebih jelas dan mudah dipahami. Dimana pada proses utama atau proses 0 pada diagram konteks dijabarkan kedalam tiga buah proses yaitu proses 1.0 (*generate* kunci dengan Blum Blum Shub), proses 2.0 (*enkripsi* CBC 256-bits), dan proses 3.0 (*dekripsi* CBC 256-bits).

1.0 (*generate* kunci dengan Blum Blum Shub) berfungsi melakukan proses pembangkitan pembangkitan kunci simteris acak sepanjang *256-bits* yang akan digunakan dalam proses enkripsi dan dekripsi. Proses ini membutuhkan masukan berupa dua buah bilangan prima  $p$  dan  $q$  yang diinputkan oleh pengirim. Kemudian hasil dari proses ini berupa kunci acak sepanjang *256-bits* yang dikirimkan ke pengirim.

Proses 2.0 (enkripsi CBC *256-bits*) berfungsi melakukan proses enkripsi dengan mode operasi CBC *256-bits* menggunakan teknik XOR. Proses ini membutuhkan dua buah data dari pengirim, yaitu kunci dan *plain text*. Hasil proses ini kemudian dikirimkan kepada penerima beserta kunci.

Proses 3.0 (dekripsi CBC *256-bits*) berfungsi melakukan proses dekripsi dari *cipher text* yang diterima dari pengirim melalui sistem. Proses ini menggunakan mode operasi CBC *256-bits* dengan teknik XOR. Proses ini membutuhkan dua buah data, yaitu: kunci dan *cipher text*. Hasil dari proses ini adalah *plain text* yang kemudian dikirimkan ke penerima pesan.

### **3.4.2. Rancangan Interface**

Interface yang akan digunakan aplikasi terbagi menjadi 2 bagian, yaitu:

#### **1. Interface Enkripsi**

Adapun keterangan dari interface di bawah adalah :

- 1) *Textarea* untuk meng-*input plain text*
- 2) Tombol untuk membangkitkan kunci sepanjang 128-bit secara acak
- 3) *Textfield* tempat dimana kunci yang berhasil dibangkitkan akan diletakkan

- 4) Tombol untuk menyimpan kunci yang berhasil dibangkitkan dalam bentuk file .txt.
- 5) Tombol untuk melakukan enkripsi *plain text* menjadi *cipher text*
- 6) *Textarea* dimana *cipher text* hasil enkripsi dari *plain text* akan diletakkan.
- 7) Tombol untuk menyimpan *cipher text* yang dihasilkan dalam bentuk file .txt.
- 8) Tombol untuk membersihkan semua form untuk menginputkan semua data baru.

Enkripsi	
Plain Text	
1	
Generate Key 2	
Kunci	3
	Simpan 4
Enkripsi 5	
Cipher Text	
6	
Simpan Cipher Text 7	
Bersih 8	

Gambar 3.4. Interface Enkripsi

Sumber: Oleh Penulis (2019)

## 2. Interface Deskripsi

Berikut merupakan interface dari form dekripsi yang akan digunakan oleh aplikasi :

**Deskripsi**

Upload Cipher Text  
Format File .txt

Cipher Text

Upload Kunci  
Format File .txt

Kunci

Plain Text

Choose File **1**

2

Choose File **3**

4

Deskripsi **5**

6

Simpan Plain Text **7**

Bersih **8**

*Gambar 3.5. Interface Deskripsi*

*Sumber : Oleh Penulis (2019)*

Adapun keterangan dari interface deskripsi yang ada di atas adalah sebagai berikut:

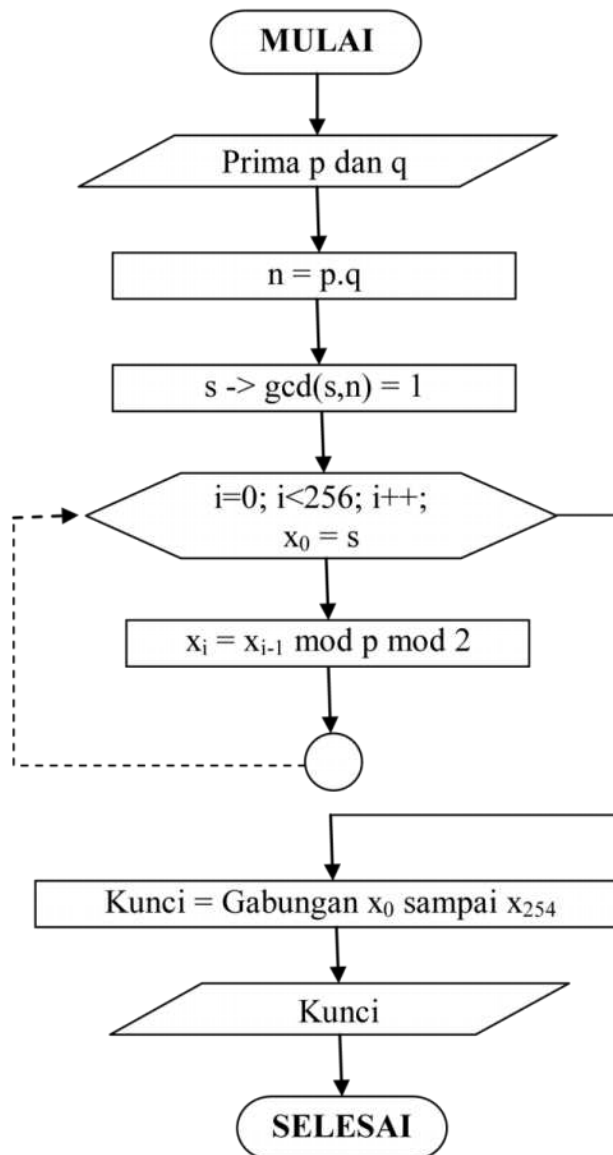
- 1) Tombol untuk mencari data *plain text* dalam bentuk file .txt untuk diupload dalam *textarea* 2.
- 2) *Textarea* dimana *cipher text* yang akan dideskripsikan akan diinputkan
- 3) Tombol untuk mencari data kunci dalam bentuk file .txt untuk diupload dalam *textfield* 4.
- 4) *Textfield* dimana kunci yang akan digunakan diinputkan
- 5) Tombol untuk melakukan deskripsi terhadap *cipher text* menjadi *plain text*
- 6) *Textarea* untuk menempatkan *plain text* hasil deskripsi *cipher text*.
- 7) Tombol untuk menyimpan *plain text* yang dihasilkan dalam bentuk file .txt
- 8) Tombol untuk membersihkan semua form untuk memulai menginputkan data baru..

### 3.4.3. Flowchart Proses

#### 1. Flowchart *Generate Kunci 256-bits*

Flowchart berikut menggambarkan proses dari pembangkitan kunci simetris sepanjang *256-bits* secara acak dengan menggunakan algoritma Blum Blum Shub.



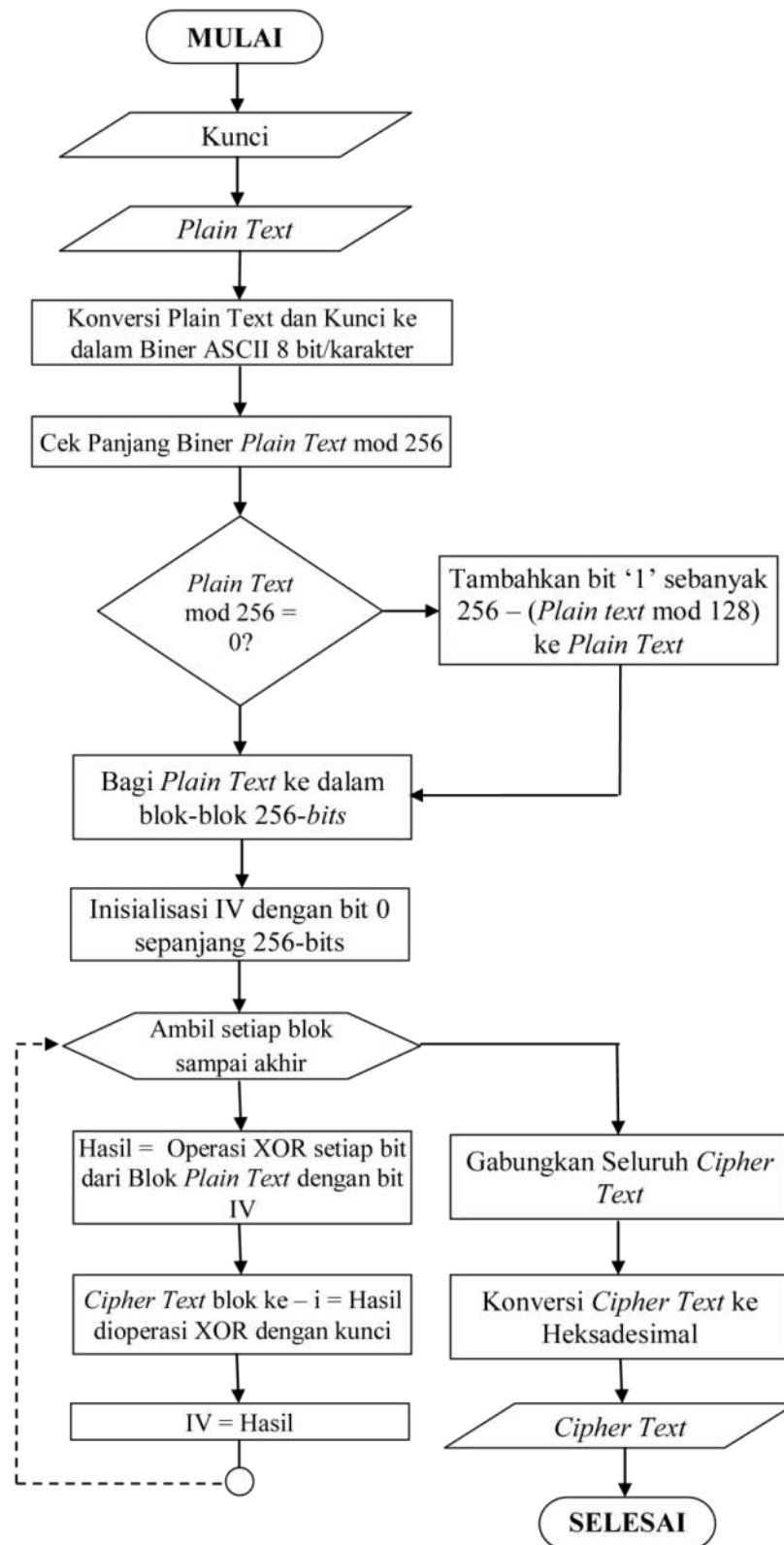


Gambar 3.6. Flowchart Generate Kunci 256-bits

Sumber: Oleh Penulis (2019)

## 2. Flowchart Enkripsi dengan Mode CBC 256-bits

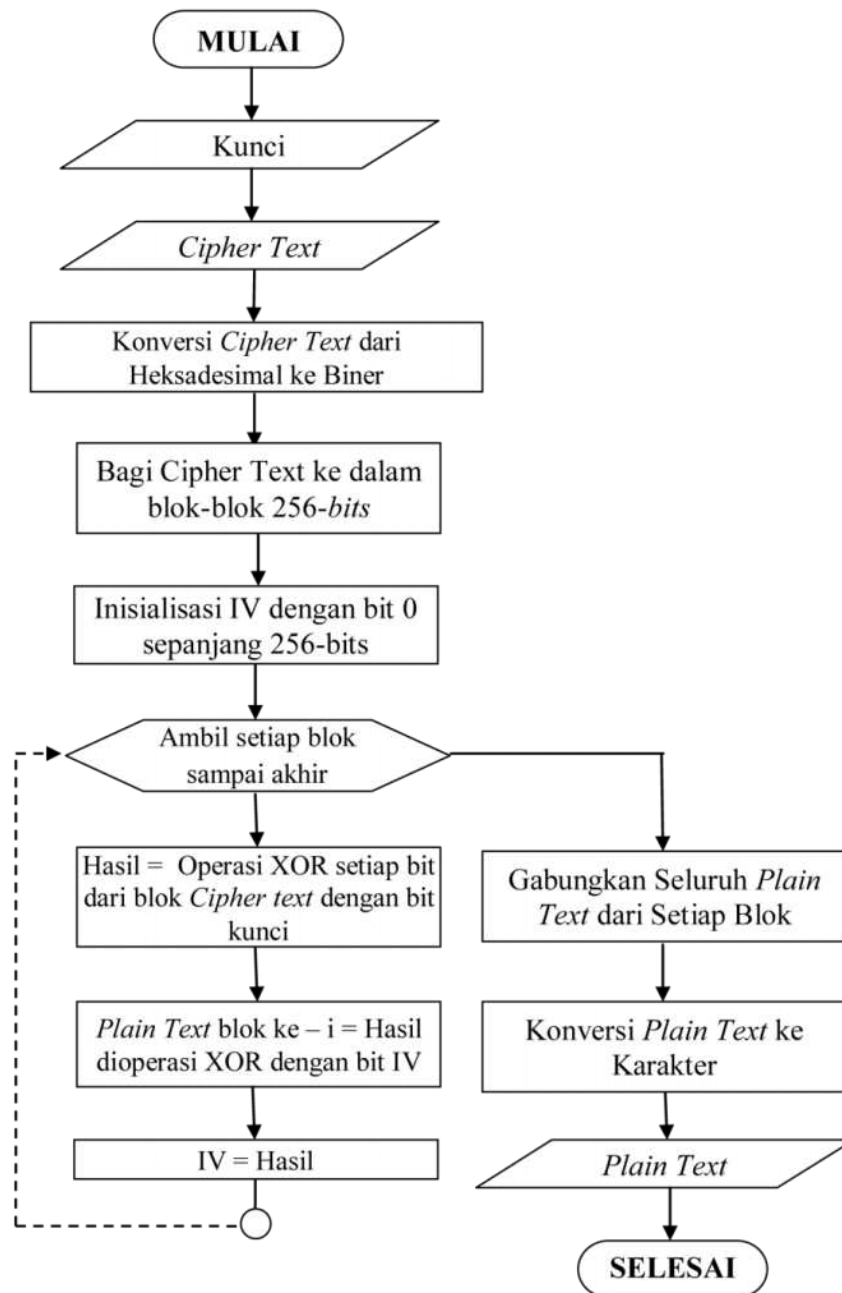
Flowchart berikut menggambarkan proses pengenkripsian *plain text* menjadi *cipher text* dengan menggunakan mode operasi CBC 256-bits dan teknik XOR.



Gambar 3.7. Flowchart Enkripsi dengan Mode CBC 256-bits  
 Sumber: Oleh Penulis (2019)

### 3. Flowchart Dekripsi dengan Mode CBC 256-bits

Flowchart berikut menggambarkan proses pendekripsian *cipher text* menjadi *plain text* dengan menggunakan mode operasi CBC 256-bits dan teknik XOR.



Gambar 3.8. Flowchart Dekripsi dengan Mode CBC 256-bits

Sumber: Oleh Penulis (2019)

## BAB IV

### HASIL DAN PEMBAHASAN

#### 4.1. Kebutuhan Spesifikasi Minimum Hardware dan Software

Setelah sistem telah berhasil dibangun dalam bentuk aplikasi, maka untuk dilakukan implementasi agar aplikasi berjalan dengan lancar, dibutuhkan beberapa perangkat yang harus tersedia, beberapa perangkat tersebut antara lain :

##### 1. Perangkat Keras

Saat melakukan pengimplementasian sistem ini untuk diuji, digunakan spesifikasi komputer sebagai berikut :

*Processor* : Intel Pentium Core i3 CPU G2030 @ 3.00 GHz

RAM : 2 GB

HD : 500 GB

Monitor : Samsung 21"

VGA : Intel (R) HD Graphics

Keyboard : Logitech

Mouse : Powerlock

Adapun perangkat minimum yang dapat digunakan agar sistem tetap dapat berjalan secara optimal adalah sebagai berikut :

*Processor* : Intel Pentium III 1.6GHz

RAM : 256 MB

HD : free 1 GB

Monitor : 19"

VGA : onboard dari motherboard

*Keyboard* : apapun

*Mouse* : apapun

## 2. Perangkat Lunak Komputer

### a. Sistem Operasi

Sistem Operasi yang digunakan dalam pengujian adalah *Windows 10 Professional* buatan pabrikan *Microsoft*. Aplikasi juga masih dapat berjalan dengan baik pada Sistem Operasi *Windows XP* dan *Windows 7*, *Windows 8*, *Windows 8.1*, dan *Windows 10*..

### b. XAMPP 1.7.3

Merupakan aplikasi yang menyediakan paket perangkat lunak yang terdiri dari beberapa perangkat lunak yang diperlukan untuk menjalankan aplikasi yang dibangun, yaitu : *Apache (web server)*, *MYSQL (database)*, *PHP (server side scripting)* dan *PhpMyAdmin*. Aplikasi masih dapat berjalan dengan baik dengan menggunakan XAMPP versi yang lebih rendah.

Aplikasi ini dibutuhkan jika pengujian dilakukan secara *local server* atau *localhost*. Sedangkan jika aplikasi telah memiliki domain sendiri dan terhubung dengan *internet*, maka aplikasi ini tidak lagi dibutuhkan, aplikasi akan dapat diakses langsung dari komputer menggunakan *browser*.

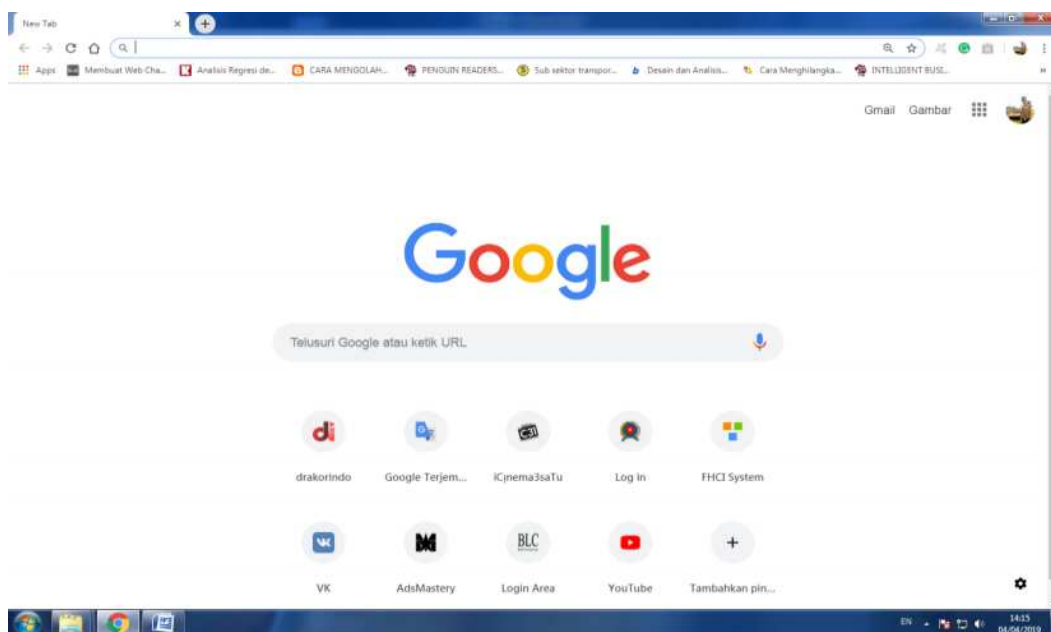
### c. Browser

Digunakan untuk mengeksekusi aplikasi, *browser* yang digunakan adalah *Google Chrome 73.0*. Aplikasi masih dapat berjalan dengan baik

menggunakan *browser* lain seperti *Mozilla Firefox*, *Internet Explore* atau *Opera*.

#### 4.2. Pengujian Aplikasi dan Pembahasan

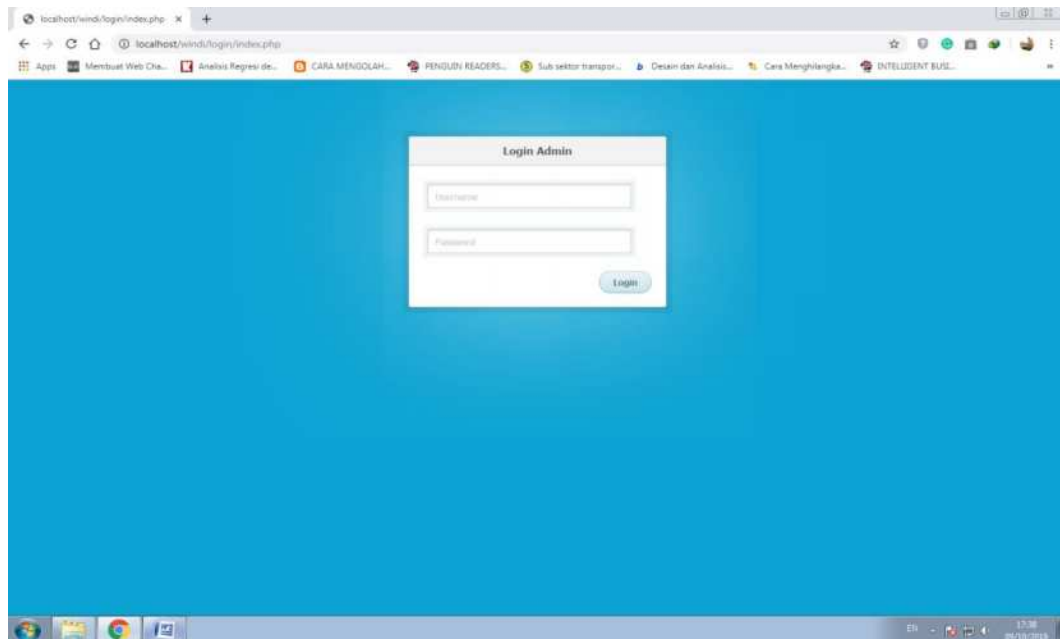
File aplikasi diletakkan ke dalam folder bayes dengan alamat file C:/xampplite/htdocs/windi/. Untuk menjalankan aplikasi, maka dibutuhkan sebuah browser, dalam pengujian yang dilakukan menggunakan browser Google Chrome. Setelah menjalankan aplikasi Google Chrome, maka tampilan Google Chrome sebagai berikut:



*Gambar 4.1. Tampilan Awal Google Chrome*

*Sumber: Hasil Pengujian Aplikasi (2019)*

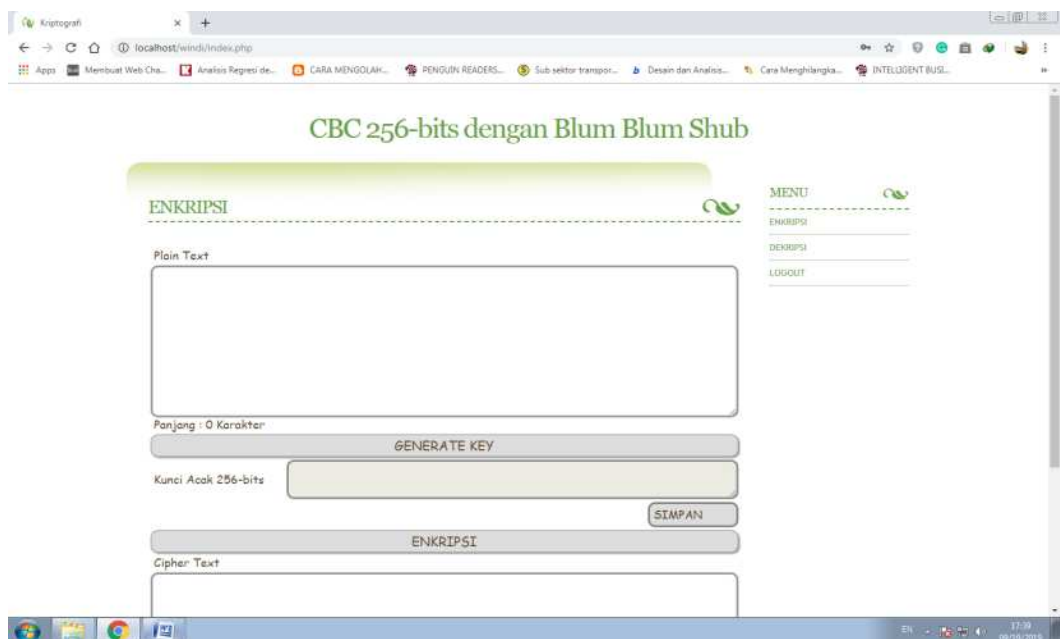
Ketikkan alamat <http://localhost/bayes> pada area *address* pada aplikasi *Google Chrome*, maka aplikasi akan ditampilkan seperti gambar berikut:



*Gambar 4.2. Tampilan Login dari Aplikasi*

*Sumber: Hasil Pengujian Aplikasi (2019)*

Untuk masuk ke dalam aplikasi, maka user harus login dengan menginput *username* dan *password* yang tepat, yaitu: *Username* : admin, *Password* : system. Kemudian klik ok, maka tampilan enkripsi dari aplikasi akan ditampilkan.



*Gambar 4.3. Tampilan Halaman Enkripsi dari Aplikasi*

*Sumber: Hasil Pengujian Aplikasi (2019)*

Pengujian yang dilakukan terbagi menjadi dua bagian, yaitu: enkripsi dan dekripsi, sedangkan analisis hasil yang dilakukan dengan menganalisis keamanan dari *cipher text* dan kunci yang dihasilkan.

#### 4.2.1. Enkripsi

Pada pengujian ini, *plain text* yang akan dijadikan sebagai contoh pesan atau data adalah teks pancasila yang berbunyi::

Pancasila

1. Ketuhanan yang Maha Esa
2. Kemanusiaan yang adil dan beradab
3. Persatuan Indonesia
4. Kerakyatan yang dipimpin oleh hikmah kebijaksanaan dan permusyawaratan perwakilan
5. Keadilan sosial bagi seluruh rakyat Indonesia.

Copy *plain text* di atas ke dalam area *plain text* pada aplikasi, lalu lalu generate key dengan mengklik tombol generate. Ketika user mengklik tombol generate key, maka aplikasi akan membangkitkan kunci acak sepanjang 256-bits ke dalam bentuk bilangan heksadesimal dan menampilkan ke dalam aplikasi. Kunci acak ini yang akan digunakan untuk mendekripsi dan dekripsi. Untuk menyimpan kunci acak tersebut, maka klik tombol simpan pada aplikasi, maka aplikasi akan menyimpan kunci yang berhasil dibangkitkan. Adapun *plain text* yang berhasil di copy ke dalam aplikasi pada area *plain text* dan kunci acak yang berhasil dibangkitkan dapat dilihat pada gambar berikut:





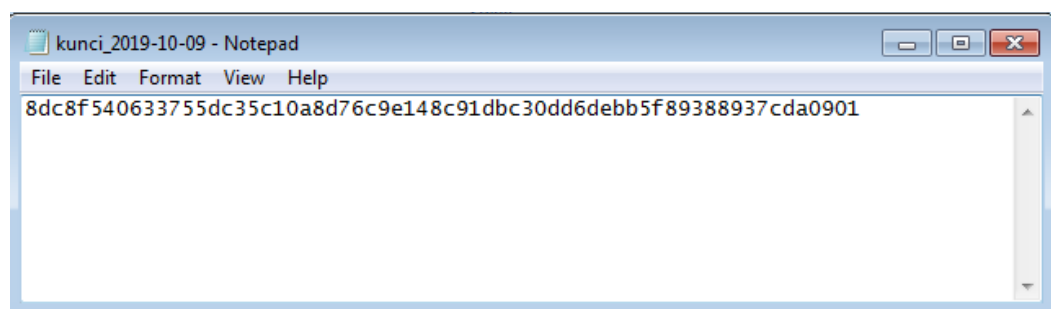
*Gambar 4.4. Tampilan Hasil Generate Key pada Aplikasi*

*Sumber: Hasil Pengujian Aplikasi (2019)*

Pada gambar di atas diketahui bahwa panjang *plain text* 231 karakter, sedangkan kunci acak 256-bits yang berhasil dibangkitkan adalah:

8dc8f540633755dc35c10a8d76c9e148c91dbc30dd6debb5f89388937cda0901

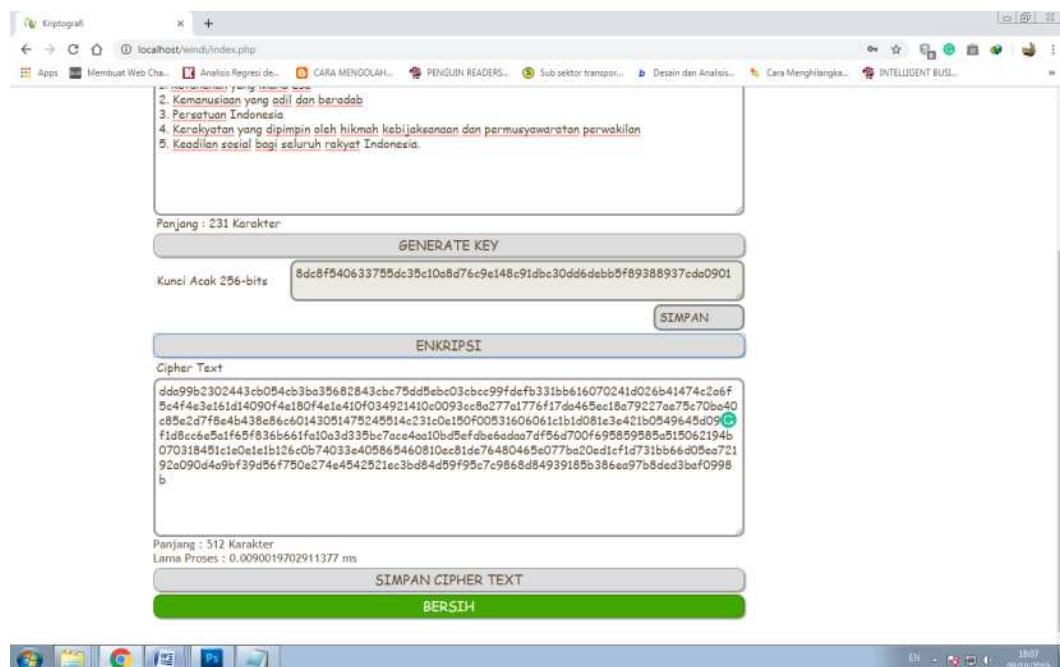
Kunci yang berhasil disimpan oleh aplikasi akan disimpan ke dalam bentuk format text yaitu: .txt, seperti pada gambar berikut:



*Gambar 4.5. Tampilan Kunci yang Berhasil Disimpan oleh Aplikasi*

*Sumber: Hasil Pengujian Aplikasi (2019)*

Untuk melakukan enkripsi, maka klik tombol enkripsi yang terdapat pada aplikasi, maka aplikasi akan melakukan enkripsi terhadap *plain text* sehingga dihasilkan *cipher text* yang akan diletakkan pada area *cipher text*. Sedangkan untuk menyimpan *cipher text* tersebut, maka klik tombol simpan *cipher text*, maka *cipher text* akan disimpan ke dalam bentuk format file .txt sama seperti menyimpan kunci sebelumnya. Hasil enkripsi yang dilakukan aplikasi dapat dilihat pada gambar berikut:



Gambar 4.6. Tampilan Hasil Hasil Enkripsi pada Aplikasi  
Sumber: Hasil Pengujian Aplikasi (2019)

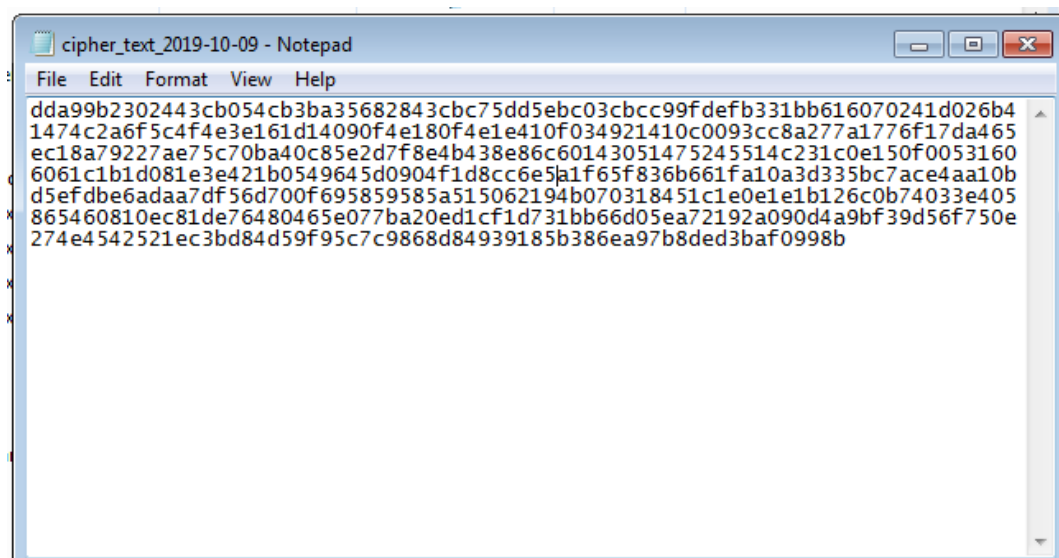
*Cipher text* hasil enkripsi terletak pada area *cipher text*, dimana *cipher text* yang dihasilkan adalah:

```
dda99b2302443cb054cb3ba35682843cbc75dd5ebc03cbcc99fdefb331bb61607024
1d026b41474c2a6f5c4f4e3e161d14090f4e180f4e1e410f034921410c0093cc8a277
a1776f17da465ec18a79227ae75c70ba40c85e2d7f8e4b438e86c601430514752455
14c231c0e150f00531606061c1b1d081e3e421b0549645d0904f1d8cc6e5a1f65f83
```

6b661fa10a3d335bc7ace4aa10bd5efdbe6adaa7df56d700f695859585a515062194b  
 070318451c1e0e1e1b126c0b74033e405865460810ec81de76480465e077ba20ed1  
 cf1d731bb66d05ea72192a090d4a9bf39d56f750e274e4542521ec3bd84d59f95c7c  
 9868d84939185b386ea97b8ded3baf0998b

Dimana *cipher text* yang dihasilkan memiliki panjang 512 Karakter dengan waktu proses yang dibutuhkan sebesar 0.0090019702911377 ms.

*Cipher text* yang berhasil disimpan oleh aplikasi akan disimpan ke dalam bentuk format text yaitu: .txt, seperti pada gambar berikut:



Gambar 4.6. Tampilan *Cipher text* yang berhasil Disimpan oleh Aplikasi

Sumber: Hasil Pengujian Aplikasi (2019)

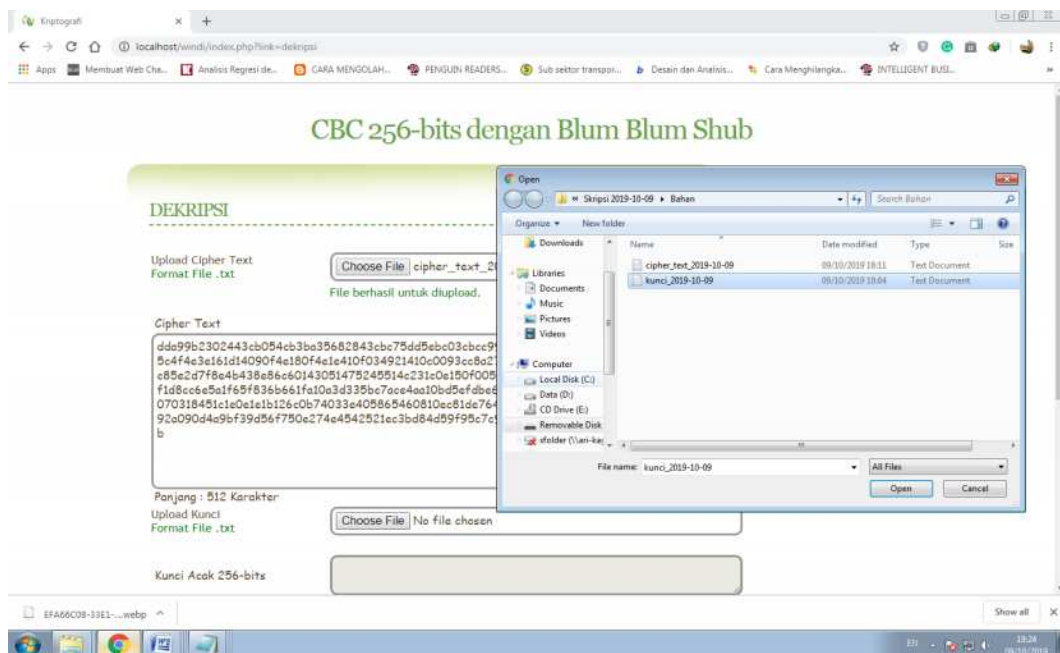
#### 4.2.2. Dekripsi

Proses dekripsi dilakukan dengan masuk terlebih dahulu ke area dekripsi pada aplikasi. Klik menu dekripsi, maka halaman dekripsi akan ditampilkan sebagai berikut:



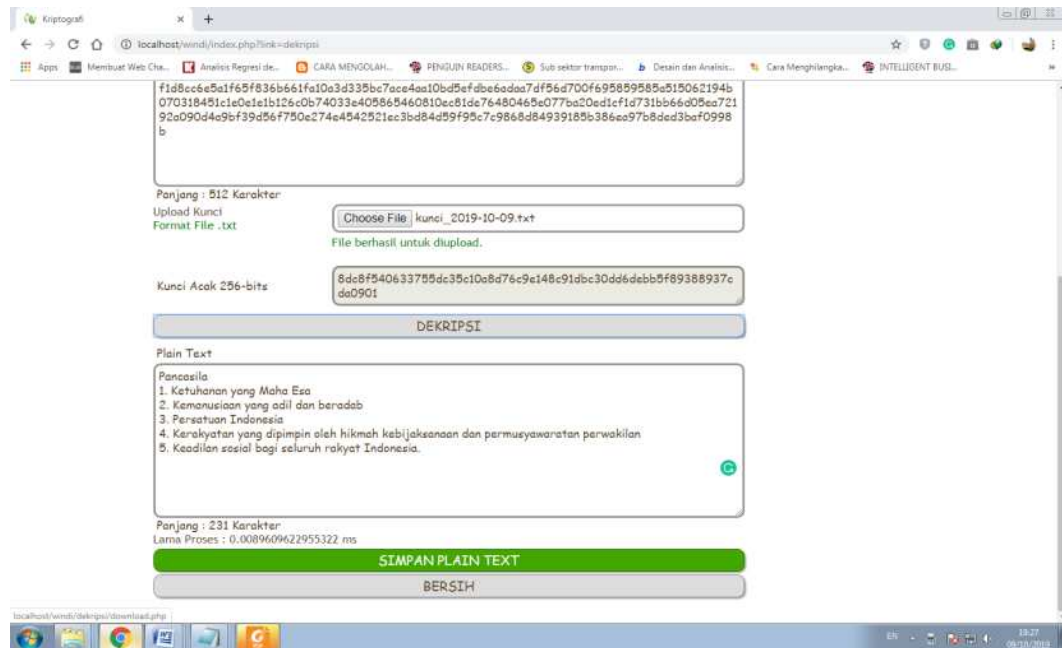
Gambar 4.7. Tampilan Halaman Dekripsi pada Aplikasi  
Sumber: Hasil Pengujian Aplikasi (2019)

Untuk mendekripsi *cipher text*, maka upload file *cipher text* dan file kunci yang berhasil diterima oleh penerima file, seperti yang ditunjukkan gambar berikut:



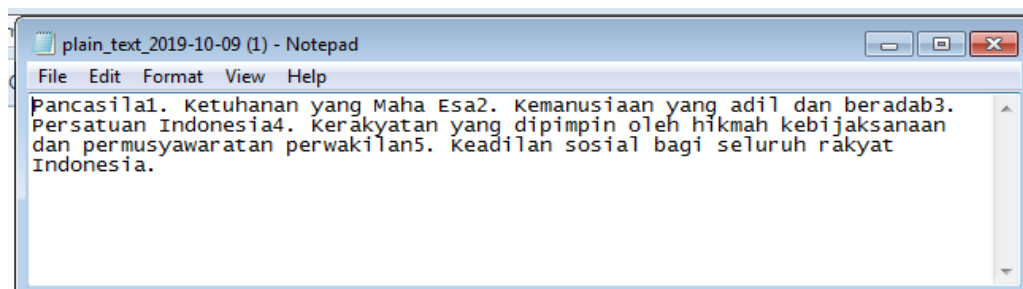
Gambar 4.8. Tampilan Upload File Cipher text dan Kunci pada Aplikasi  
Sumber: Hasil Pengujian Aplikasi (2019)

Untuk mendekripsi *cipher text* maka klik tombol dekripsi pada aplikasi, maka aplikasi akan melakukan proses dekripsi terhadap *cipher text* dengan menggunakan kunci yang diinput user dan meletakkan *plain text* hasil dekripsi pada area *plain text* seperti yang ditunjukkan pada gambar.



Gambar 4.9. Tampilan Hasil Dekripsi pada Aplikasi  
Sumber: Hasil Pengujian Aplikasi (2019)

Pada gambar di atas, pesan asli atau *plain text* berhasil dibentuk kembali, dimana waktu proses untuk dekripsi sebesar: 0.0089609622955322 ms. Untuk menyimpan *plain text* yang diperoleh, maka klik simpan *plain text*, maka aplikasi akan menyimpan *plain text* ke dalam format file .txt seperti gambar di bawah ini:



Gambar 4.10. Tampilan Plain text yang Berhasil Disimpan oleh Aplikasi  
Sumber: Hasil Pengujian Aplikasi (2019)

### **4.3. Analisis Keamanan**

Analisis keamanan terbagi menjadi dua bagian analisis, yaitu analisis keamanan pada *cipher text* dan analisis keamanan pada kunci.

#### **4.3.1. Analisis Keamanan *Cipher text***

*Cipher text* merupakan hasil enkripsi dengan mode operasi *Cipher Block Chaining* (CBC) dengan panjang blok 256-bits, sehingga *plain text* akan dipecah ke dalam blok 256-bits dan akan dihasilkan blok-blok acak sepanjang 256-bits. Blok berikutnya akan berhubungan langsung dengan blok sebelumnya, sehingga perubahan *cipher text* yang terjadi pada blok sebelumnya akan merusak seluruh rantai dari *cipher text* setelahnya walau hanya perubahan sebanyak 1 bit *cipher text*. Hal ini membuat keaslian *plain text* tetap terjaga. Selain itu, hal ini juga membuat blok sebelumnya dengan blok setelahnya tidak memiliki pola yang sama sehingga *cipher text* mustahil bisa dipecahkan dengan teknik frekuensi, dikarenakan blok *plain text* yang sama tidak akan menghasilkan *cipher text* yang seragam. Sehingga cara yang memungkinkan untuk mendekripsi *cipher text* adalah dengan menggunakan kunci enkripsi-dekripsi yang digunakan sebelumnya.

#### **4.3.2. Analisis Keamanan Kunci**

Kunci enkripsi-dekripsi yang digunakan dibangkitkan dengan acak menggunakan algoritma Blum Blum Shub. Hal ini membuat kunci yang dibangkitkan memiliki keacakan yang sangat baik dan sulit ditebak maupun diprediksi. Keacakan kunci ini membuat intruder harus mencoba seluruh kombinasi kunci untuk mendapatkan kunci yang benar yang dapat digunakan untuk mendekripsi *cipher text*. Dengan panjang kunci sepanjang 256-bits, maka

kemungkinan kunci yang dapat dibangkitkan adalah  $2^{256} = 1,1579 \times 10^{77}$ . Jika diasumsikan bahwa intruder mampu mencoba 100 triliun atau  $10^{14}$  kunci per detik dengan teknik *bruto force* (mencoba semua kemungkinan kunci satu per satu) untuk menemukan kunci yang benar, maka dibutuhkan:

$$\begin{aligned} \text{time} &= \frac{2^{256}}{10^{14}} = 1,1579 \times 10^{63} \text{ detik} \\ &= 3,6717 \times 10^{49} \text{ Juta Tahun} \end{aligned}$$

Besarnya waktu yang dibutuhkan intruder untuk memecahkan kunci yang digunakan membuat kunci menjadi sangat aman dan layak untuk digunakan.

#### **4.4. Kelebihan dan Kekurangan Algoritma yang Dirancang**

Algoritma yang dirancang memiliki beberapa kelebihan dan kekurangan, yaitu:

##### **4.4.1. Kelebihan**

Algoritma yang dirancang memiliki kecepatan proses yang baik, dimana waktu yang dibutuhkan untuk proses enkripsi dan dekripsi sangat cepat sehingga memiliki kompleksitas algoritma yang rendah. Selain itu, algoritma yang dirancang juga hemat dalam hal sumber daya yang digunakan karena prosesnya yang sederhana sehingga tidak membutuhkan spesifikasi perangkat keras yang tinggi. Dari segi keamanan, *cipher text* yang dihasilkan juga dikategorikan sangat aman, selain itu kunci acak yang digunakan juga sangat aman dan sangat sulit dipecahkan dengan teknik *bruto force*.

#### 4.4.2. Kekurangan

Kunci acak untuk enkripsi-dekripsi yang digunakan adalah sama atau termasuk ke dalam kunci simetris. Hal ini membuat terdapat masalah *key distribution*, dimana tidak ada cara yang benar-benar aman untuk mengirimkan kunci dari pengirim pesan kepada penerima pesan. Hal ini dikarenakan penerima pesan membutuhkan kunci tersebut untuk melakukan dekripsi, namun kunci tersebut dibangkitkan oleh pengirim pesan sehingga penerima pesan harus mengirimkan kunci tersebut kepada penerima pesan. Kendala yang terjadi bahwa kunci acak yang digunakan sangat rahasia, sehingga jika kunci berhasil disadap maka *cipher text* akan terbongkar. Jika kunci dikirimkan oleh pengirim pesan kepada penerima pesan baik dari jalur komunikasi apapun, maka kunci acak tersebut berpeluang sangat besar untuk disadap. Sehingga menjaga kerahasiaan kunci menjadi sangat fatal dan fatal saat pendistribusian kunci dari pengirim pesan ke penerima pesan. Sehingga dibutuhkan mekanisme lain untuk mengatasi masalah *key distribution* ini.



## BAB V

### KESIMPULAN DAN SARAN

#### 5.1. Kesimpulan

Berdasarkan pengujian sistem yang telah dilakukan, maka dapat diambil beberapa kesimpulan sebagai berikut:

1. Keamanan data atau pesan dapat ditingkatkan dengan proses enkripsi menggunakan teknik XOR dengan mode operasi Cipher Block Chaining (CBC) 256-bits dan kunci acak Blum Blum Shub, dimana waktu yang dibutuhkan untuk memecahkan kunci acak yang digunakan hingga  $3,6717 \times 10^{49}$  Juta Tahun dan *cipher text* sangat sensitif terhadap perubahan sehingga keaslian *cipher text* juga terjaga karena setiap blok *cipher text* selalu berkaitan dengan blok sebelumnya. Setiap blok yang sama akan menghasilkan blok *cipher text* yang bervariasi sehingga *cipher text* aman dari analisis frekuensi.
2. Kombinasi mode operasi CBC, Teknik XOR, dan Algoritma Blum Blum Shub Mode dapat dilakukan dengan cara mode operasi CBC 256-bits digunakan sebagai mode operasi enkripsi-dekripsi dimana proses enkripsi dilakukan dengan teknik XOR dan kunci dibangkitkan dengan algoritma Blum Blum Shub.
3. Kombinasi operasi XOR, mode operasi CBC dengan Kunci Acak Blum Blum Shub dapat diimplementasikan kedalam program aplikasi, dimana pada penelitian ini diimplementasikan dengan bahasa pemrograman PHP. Disaat kombinasi algoritma ini diimplementasikan, waktu proses yang

dibutuhkan baik untuk enkripsi dan dekripsi sangat cepat, dan perangkat untuk menjalankan aplikasi juga sangat ringan, sehingga algoritma yang dirancang hemat dalam hal sumber daya komputer dan waktu proses.

## 5.2. Saran

Beberapa saran yang diusulkan untuk mengembangkan penelitian ini menjadi lebih baik adalah sebagai berikut:

1. Disarankan untuk melakukan penelitian lebih lanjut untuk mengatasi masalah *key distribution* yang ada pada kombinasi algoritma yang dirancang pada penelitian ini.
2. Disarankan untuk menambah panjang blok *cipher text* yang digunakan dari 256-bits menjadi 512-bits untuk lebih meningkatkan keamanan pesan dan kunci acak yang digunakan.

## DAFTAR PUSTAKA

- Ariyus, D. (2017). *Pengantar Ilmu Kriptografi*. Yogyakarta: Andi Offset.
- Anraeni, S., Herdianti., dan Mursyid. (2016). *Hybrid Methods of Ciphertext and RSA Cryptographic Algorithm Using Classical Vigenère*. International Journal of Computing and Informatics (IJCANDI) ISSN: 2502-2334 Vol 1, No 2, May 2016, pp. 68-74
- Andrian, Yudhi, And Purwa Hasan Putra. "Analisis Penambahan Momentum Pada Proses Prediksi Curah Hujan Kota Medan Menggunakan Metode Backpropagation Neural Network." Seminar Nasional Informatika (Snif). Vol. 1. No. 1. 2017.
- Batubara, Supina. "Analisis Perbandingan Metode Fuzzy Mamdani Dan Fuzzy Sugeno Untuk Penentuan Kualitas Cor Beton Instan." It Journal Research And Development 2.1 (2017): 1-11.
- Dhany, H. W., Izhari, F., Fahmi, H., Tulus, M., & Sutarman, M. (2017, October). Encryption And Decryption Using Password Based Encryption, Md5, And Des. In International Conference On Public Policy, Social Computing And Development 2017 (Icoposdev 2017) (Pp. 278-283). Atlantis Press.
- Dashti, A., Kheradmand, H. A., & Jazi, M. D. (2016). *Comparison Of Three Modes Of Cryptography Operation For Providing Security and Privacy Based on Important Factors*. International Journal of Information Technology and Electrical Engineering Volume 5, Issue 3 ISSN: - 2306-708X June 2016
- Fuad, R. N., & Winata, H. N. (2017). Aplikasi Keamanan File Audio Wav (Waveform) Dengan Terapan Algoritma Rsa. Infotekjar: Jurnal Nasional Informatika Dan Teknologi Jaringan, 1(2), 113-119.
- Ginting, G., Fadlina, M., Siahaan, A. P. U., & Rahim, R. (2017). Technical Approach Of Topsis In Decision Making. Int. J. Recent Trends Eng. Res, 3(8), 58-64.
- Hafni, Layla, And Rismawati Rismawati. "Analisis Faktor-Faktor Internal Yang Mempengaruhi Nilai Perusahaan Pada Perusahaan Manufaktur Yang Terdaftar Di Bei 2011-2015." Bilancia: Jurnal Ilmiah Akuntansi 1.3 (2017): 371-382.
- Hartanto, S. (2017). Implementasi Fuzzy Rule Based System Untuk Klasifikasi Buah Mangga. Techsi-Jurnal Teknik Informatika, 9(2), 103-122.
- Hariyanto, E., Lubis, S. A., & Sitorus, Z. (2017). Perancangan Prototipe Helm Pengukur Kualitas Udara. Komik (Konferensi Nasional Teknologi Informasi Dan Komputer), 1(1).
- Iqbal, M., Siahaan, A. P. U., Purba, N. E., & Purwanto, D. (2017). Prim's Algorithm For Optimizing Fiber Optic Trajectory Planning. Int. J. Sci. Res. Sci. Technol, 3(6), 504-509.

- Jogiyanto. (2015). *Pengenalan Komputer*. Yogyakarta: Andi Offset.
- Kurnia, D., Dafitri, H., & Siahaan, A. P. U. (2017). Rsa 32-Bit Implementation Technique. *Int. J. Recent Trends Eng. Res*, 3(7), 279-284.
- Kromodimoeljo, S. (2014). *Teori dan Aplikasi Kriptografi*. Jakarta; SPK IT Consulting.
- Kumar, G., Rai, M., & Lee, G. S. (2015). *Implementation of Cipher Block Chaining in Wireless Sensor Networks for Security Enhancement*. *International Journal of Security and Its Applications* Vol. 6, No. 1, January, 2015
- Kumar, S., Suneetha, C. H., & Chandrasekhar, A. (2016). *A Block Cipher Using Rotation and Logical XOR Operations*. *IJCSI International Journal of Computer Science Issues*, Vol. 8, Issue 6, No 1, November 2011 ISSN (Online): 1694-0814
- Ladjamudin, A. B. (2016). *Rekayasa Perangkat Lunak*. Yogyakarta: Graha Ilmu. Ladjamudin, A. B. (2018). *Analisa dan Desain*. Yogyakarta: Graha Ilmu.
- Lietara, A. P. (2016). *Analisis & Perbandingan Blum Blum Shub dan Inversive Congruential Generator Beserta Implementasinya*. *Jurnal ITB*.
- Marlina, L., Putera, A., Siahaan, U., Kurniawan, H., & Sulistianingsih, I. (2017). Data Compression Using Elias Delta Code. *Int. J. Recent Trends Eng. Res*, 3(8), 210-217.
- Madcoms. (2015). *Mahir dalam 7 Hari Adobe Dreamweaver CS 6 dengan Pemrograman PHP & MYSQL*. Yogyakarta: Andi Offset. Munir, R. (2016). *Kriptografi*. Bandung: Informatika.
- Pascal, B. (2017). *Handbook Jaringan Komputer dan Keamanannya*. Bandung: Informatika
- Putri, R. E., & Siahaan, A. (2017). Examination Of Document Similarity Using Rabin-Karp Algorithm. *International Journal Of Recent Trends In Engineering & Research*, 3(8), 196-201.
- Rachman, A. K. (2015). *Perbandingan Mode Chiper Electronic Code Book Dan Chiper Block Chaining Dalam Pengamanan Data*. *Jurnal Teknologi*, Volume 3 Nomor 1 , Juni 2015, 84-89
- Rosmala, D., dan Aprian, R. (2015). *Implementasi Mode Operasi Cipher Block Chaining (CBC) Pada Pengamanan Data*. *Jurnal Informatika* No.2 , Vol. 3, Mei – Agustus 2015
- Sari, R. D., Supiyandi, A. P. U., Siahaan, M. M., & Ginting, R. B. (2017). A Review Of Ip And Mac Address Filtering In Wireless Network Security. *Int. J. Sci. Res. Sci. Technol*, 3(6), 470-473.
- Sarif, M. I. (2017). *Penemuan Aturan Yang Berkaitan Dengan Pola Dalam Deret Berkala (Time Series)*.

- Syahputra, Rizki, And Hafni Hafni. "Analisis Kinerja Jaringan Switching Clos Tanpa Buffer." *Journal Of Science And Social Research* 1.2 (2018): 109-115.
- Schneier, B. (2016). *Applied Cryptography: Protocols, Algorithms and Source Code in C. 6th Edition*. New Jersey: John Wiley & Sons, Inc.
- Sridevi. (2014). *Construction of Stream Ciphers from Block Ciphers and their Security*. *IJCSMC*, Vol. 3, Issue. 9, September 2014, pg.703 – 714
- Subandi, A. (2017). *Three-Pass Protocol Implementation in Vigenere Cipher Classic Cryptography Algorithm with Keystream Generator Modification*. *Advances in Science, Technology and Engineering Systems Journal (ASTES)*
- Sanjaya, M. B. (2017). *Perancangan dan Implementasi Random Number Blum Blum Shub pada Dynamic Cell Spreading untuk Pengamanan Berkas*. Volume.1 - November 2017. Seminar Nasional Multi Disiplin Ilmu p-ISSN = 2598-4969 e- ISSN = 2598-5191