Aplikasi Enkripsi Sederhana untuk Penyimpanan *Database* Password

Muhammad Reza Ramadhan - 13514107
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
rezaramadhan.m@students.itb.ac.id

***Abstract*—Penggunaan enkripsi pada metode penyimpanan password adalah salah satu cara yang mudah namun memiliki beberapa resiko. Makalah ini secara garis besar membahas teori-teori dasar enkripsi sejarah dan pengertian istilah-istilah enkripsi, bagaimana implementasi enkripsi pada metode penyimpanan password, salah satu contoh algoritma enkripsi yang bisa digunakan, serta kelebihan dan kekurangan dari algoritma enkripsi tersebut dan metode penggunaan enkripsi ini secara umum.**

***Kata Kunci*—Ciphertext, enkripsi, modulo, password, plaintext.**

I. PENDAHULUAN

Media sosial, forum, surat elektronik, serta blog adalah salah satu contoh aplikasi berbasis web yang membutuhkan kombinasi password dan username tertentu. Pada beberapa situs tertentu seperti kaskus.co.id , terdapat ribuan kombinasi password serta username baru yang dibuat setiap harinya. Setiap akun yang ada suatu hal yang bersifat sangat privasi. Karena itulah diperlukan sebuah metode penyimpanan kombinasi password serta username yang tersembunyi dengan baik sehingga tidak mudah diakses oleh orang lain yang tidak berkepentingan.

Pada umumnya, hampir setiap situs menggunakan teknik yang sama untuk penyimpanan username, yaitu menggunakan plaintext (teks biasa tanpa diubah) sehingga mudah dilihat oleh orang lain. Namun, terdapat beberapa metode penyimpanan password, mulai dari disimpan dalam plaintext, penyimpanan password setelah sebelumnya dienkripsi dengan cara tertentu, serta penggunaan fungsi hash dalam penyimpanan password.

Penyimpanan password menggunakan plaintext adalah sebuah cara yang sangat ceroboh karena jika seseorang telah berhasil memasuki server dengan cara tertentu − sebutlah *hacker* untuk mempermudah − ia bisa dengan mudah mengambil data username dan password dari semua user yang terdaftar. Karena itulah penggunaan cara penyimpanan lain sangat dianjurkan untuk mencegah peristiwa-peristiwa tidak diinginkan yang mungkin terjadi.

Dengan menggunakan enkripsi, penyimpanan password menjadi sedikit lebih aman, karena password telah disimpan dalam bentuk terenkripsi dan tidak mudah bagi hacker untuk mendapatkan data password yang sebenarnya.

II. DASAR TEORI

A. *Aritmetika Modulo*

Salah satu cabang dari matematika yang sering digunakan dalam ilmu komputasi dan aplikasinya dalam enkripsi sebuah data, baik data berupa teks ataupun lainnya adalah Aritmetika modulo. Aritmetika modulo berfokus pada operator **mod.**

Secara singkat, fungsi dari operator mod adalah mencari sisa pembagian dari dua bilangan bulat. Sehingga dapat dinyatakan bahwa:

Misalkan a adalah bilangan bulat dan m adalah bilangan bulat positif. Operasi a mod m (dibaa "a modulo m") memberikan sisa jika a dibagi dengan m. Dengan kata lain: Jika $a \bmod m = r$ sedemikian sehingga $a = mq + r,\ dengan\ 0 \leq r < m$ (Munir, 2006:V-13)

Dalam ekspresi a mod m, setiap bilangan bulat m pasti memiliki beberapa nilai a berbeda yang membuat ekspresi a mod m tersebut masih bernilai sama. Salah satu contohnya adalah ketika m = 5, maka nilai a = 1, 6, 11, 16, 21, 26, … akan membuat nilai a mod m bernilai sama yaitu 1. Pasangan nilai a berbeda yang membuat ekpresi modulo a mod m bernilai sama disebut kekongruenan.

Sehingga menurut Munir (2006:V-14): misalkan a dan b adalah bilangan bulat dan m adalah bilangan positif, maka a ≡ b (mod m) jika dan hanya jika m habis membagi a dan b.

Dalam aritmetika modulo juga mengenal sebuah istilah bernama invers modulo, yaitu jika a dan m relatif prima dan m > 1, maka terdapat sebuah invers modulo dari a mod m, yang dapat dicari dengan

$$a\bar{a} \equiv mod\ m$$

Dengan $\bar{a}$ menyatakan invers modulo dari a terhadap m. Pencarian invers modulo dapat dicari dengan fakta bahwa jika a dan m relatif prima, maka bilangan pembagi bersama terbesar dari keduanya adalah satu, sehingga terdapat persamaan

$$pa + qm = 1$$

Yang mengimplikasikan

$$pa \equiv 1 \ (mod \ m)$$

Terlihat bahwa nilai p pada persamaan diatas adalah hasil dari invers modulo dari *a* mod *m*.

B. Enkripsi

Enkripsi data sudah dikenal oleh umat manusia sejak tahun 1900 SM, yang digunakan oleh bangsa mesir dalam penggunaan hieroglif yang berbeda dari hieroglif biasa untuk menyembunyikan makna yang sebenarnya dari suatu tulisan rahasia.

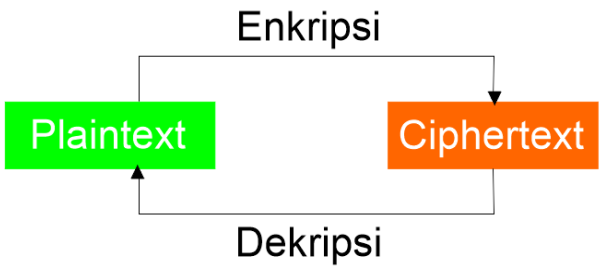
Enkripsi berasal dari bahasa Yunani kryptos yang berarti tersembunyi atau rahasia (Rouse, techtarget.com). Secara sederhana enkripsi adalah proses merubah data sedemikian rupa sehingga data tersebut tidak mudah dibaca oleh sembarang orang. Tidak sedikit jenis enkripsi yang ada di dunia ini, mulai dari enkripsi paling sederhana yaitu mengubah suatu alfabet menjadi simbol aneh seperti sandi morse, menggeser setiap alfabet sekian kali sehingga menjadi alfabet baru, mesin Enigma yang digunakan oleh tentara Nazi pada Perang Dunia kedua, hingga algoritma RSA yang banyak digunakan oleh bank-bank di seluruh dunia untuk mengenkripsi data nasabahnya.

Selain enkripsi, terdapat pula yang disebut dekripsi yaitu proses pengembalian data yang telah dienkripsi menjadi data biasa yang dapat dibaca oleh semua orang. Proses dekripsi ini menggunakan algoritma yang merupakan kebalikan dari algoritma enkripsi. Contohnya adalah pada algoritma penggeseran alfabet, maka untuk mendekripsi hal tersebut maka diperlukan penggeseran kebalikan dari penggeseran yang dilakukan diawal.

Data awal yang belum dienkripsi biasa disebut plaintext, sedangkan data akhir yang telah dienkripsi biasa disebut ciphertext. Algoritma-algoritma yang digunakan untuk merubah plaintext ini menjadi ciphertext sendiri lah yang membuat dunia enkripsi menjadi sangat beragam.

Pada dunia ilmu komputasi terdapat sebuah cabang ilmu bernama kriptografi, yaitu ilmu yang mempelajari cara menjaga kerahasiaan pesan dengan cara menyamarkannya menjadi bentuk tersandi yang tidak mempunyai makna (Munir, 2006:V-21).

Berikut adalah ilustrasi mengenai proses perubahan informasi pada alur enkripsi-dekripsi:



Gambar 2.1: Ilustrasi Enkripsi-Dekripsi

Salah satu contoh aplikasi enkripsi pada kehidupan sehari-hari adalah enkripsi pada program televisi yang disebar oleh penyedia televisi kabel. Sebelum data program televisi tersebut disebar, program televisi tersebut dienkripsi dengan cara tertentu. Setelah data terenkripsi tersebut diterima oleh pelanggan, *decoder* milik masing-masing pelanggan yang telah dibeli sebelumnya akan mendekripsi sinyal terenkripsi tadi sehingga program televisi tersebut bisa disaksikan kembali oleh pelanggan.

III. GAMBARAN PENGGUNAAN

A. Gambaran Proses Penyimpanan Password

Pada umumnya, terdapat beberapa proses dasar yang pasti terjadi ketika sebuah pengguna baru mendaftarkan username dan password miliknya kepada sebuah situs. Username pengguna langsung dikirimkan pada server tanpa enkripsi apapun dan langsung disimpan begitu saja pada server. Sedangkan pada metode penggunaan enkripsi dalam penyimpanan password ini sebelum password disimpan dalam server password akan dienkripsi dengan algoritma tertentu yang telah dibuat sebelumnya. Berikut adalah salah satu contoh bagaimana perbandingan data username dan password setiap user dan penyimpanan data username dan password pada suatu server yang menggunakan metode enkripsi:

Data sebenarnya			**Data pada server**	
Username	Password		Username	Password
muhr	rumahku		muhr	VR9DZ 7E0Uh8=
Reza32	inipassword		Reza32	OPusiP/ blQ642k CalyuoP Q==
1rmdhn2	123abcd		1rmdhn2	BWKM YK3T +B8=
m.reza.r	asdfqwert		m.reza.r	V7Q52V BKBsZF HnO7h0 fyMw==

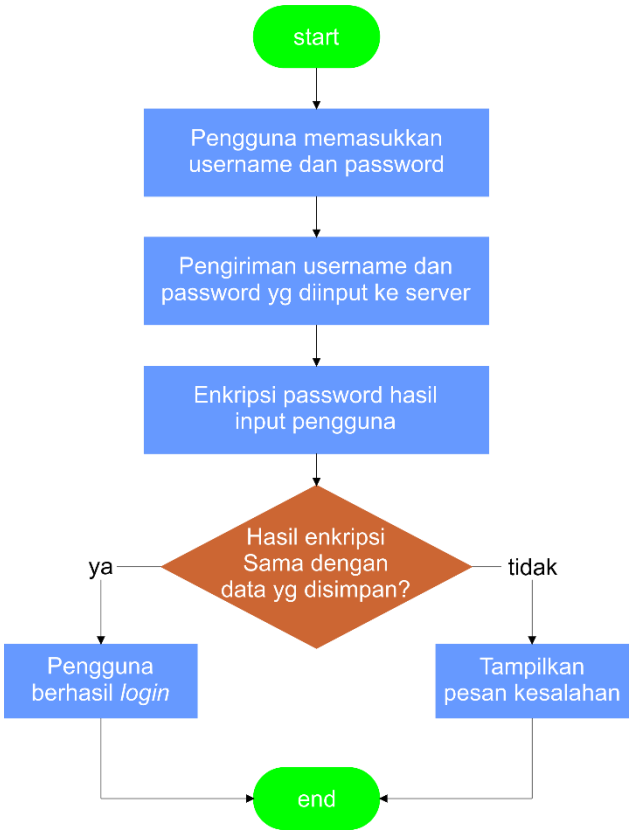
Tabel 3.1 : Perbandingan data pada server dan data sebenarnya dengan tool enkripsi pada http://textmechanic.com dengan key 111

Terlihat bahwa semua data password yang disimpan pada server tidak mudah dibaca dan jika seseorang berhasil masuk kedalam jaringan server dan menemukan file berisi data username dan password maka ia tidak akan langsung mendapatkan data sensitif ini melainkan perlu mencari algoritma enkripsi khusus yang dapat digunakan untuk menemukan plaintext yang sebenarnya dari data password tersebut.

B. Gambaran Proses Validasi saat Pengguna Log In

Pada saat pengguna melakukan *log in* pada layanan yang kita buat, hal pertama yang dilakukan adalah mengirimkan data username dan password tersebut pada server, kemudian server akan mengenkripsi password tersebut dan membandingkannya dengan hasil enkripsi yang telah disimpan oleh user. Jika hasilnya sama maka input

password tersebut valid dan jika tidak maka input password tersebut tidak valid. Berikut adalah ilustrasi flowchart validasi password user:



Gambar 3.1: Flowchart proses validasi password pengguna

Metode ini mengandung resiko bahwa mungkin saja terdapat seseorang yang telah merekayasa proses pengiriman password sehingga data tersebut dikirimkan pada server lain terlebih dahulu lalu dikirimkan ke server sebenarnya. Akan tetapi, metode pengenkripsian di server ini lebih aman dibandingkan dengan ketika enkripsi dilakukan di web lalu hasil enkripsi dikirimkan. Metode kedua menyebabkan seseorang bisa dengan mudah mengetahui fungsi yang kita gunakan untuk mengenkripsi password user, sehingga enkripsi kita yang seharusnya aman tidak menjadi aman lagi.

Terdapat satu hal yang perlu diperhatikan dalam penulisan pesan kesalahan ketika input user tidak valid, yaitu kita hanya perlu menampilkan pesan kesalahan seperti "*invalid username or password*" daripada pesan kesalahan seperti *"invalid password for username XXX"*. Penampilan tipe pesan kesalahan seperti yang pertama membuat cara *bruteforce* dan *dictionary attack* lebih susah karena pelaku *bruteforce* atau *dictionary attack* tidak tahu apakah username atau password yang tidak valid dari kombinasi yang telah ia lakukan.

C. Contoh Algoritma Enkripsi Sederhana

Pada bagian ini penulis akan mencoba membuat sebuah algoritma sederhana untuk enkripsi dengan memanfaatkan teks dengan enkoding ASCII (American Standard Code for Information Interchange). Algoritma buatan penulis ini secara singkat akan mengubah setiap karakter input menjadi representasi desimal dari karakter tersebut pada ASCII lalu "menggeser" representasi desimal tersebut sejauh jarak tertentu lalu merubah hasil pergeseran tersebut mejadi karakter ASCII kembali. Sebagai contoh, misalnya untuk karakter 'A' akan diubah menjadi 65, lalu digeser sejauh KEY sebesar 30. Representasi desimal itu kemudian akan berubah menjadi 95, dan karakter dengan representasi desimal 95 di tabel ASCII adalah '_'.

Sebagai bahan referensi, berikut adalah tabel ASCII:

Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char
0	00	Null	32	20	Space	64	40	@	96	60	`
1	01	Start of heading	33	21	!	65	41	A	97	61	a
2	02	Start of text	34	22	"	66	42	B	98	62	b
3	03	End of text	35	23	#	67	43	C	99	63	c
4	04	End of transmit	36	24	$	68	44	D	100	64	d
5	05	Enquiry	37	25	%	69	45	E	101	65	e
6	06	Acknowledge	38	26	&	70	46	F	102	66	f
7	07	Audible bell	39	27	'	71	47	G	103	67	g
8	08	Backspace	40	28	(	72	48	H	104	68	h
9	09	Horizontal tab	41	29	)	73	49	I	105	69	i
10	0A	Line feed	42	2A	*	74	4A	J	106	6A	j
11	0B	Vertical tab	43	2B	+	75	4B	K	107	6B	k
12	0C	Form feed	44	2C	,	76	4C	L	108	6C	l
13	0D	Carriage return	45	2D	-	77	4D	M	109	6D	m
14	0E	Shift out	46	2E	.	78	4E	N	110	6E	n
15	0F	Shift in	47	2F	/	79	4F	O	111	6F	o
16	10	Data link escape	48	30	0	80	50	P	112	70	p
17	11	Device control 1	49	31	1	81	51	Q	113	71	q
18	12	Device control 2	50	32	2	82	52	R	114	72	r
19	13	Device control 3	51	33	3	83	53	S	115	73	s
20	14	Device control 4	52	34	4	84	54	T	116	74	t
21	15	Neg. acknowledge	53	35	5	85	55	U	117	75	u
22	16	Synchronous idle	54	36	6	86	56	V	118	76	v
23	17	End trans. block	55	37	7	87	57	W	119	77	w
24	18	Cancel	56	38	8	88	58	X	120	78	x
25	19	End of medium	57	39	9	89	59	Y	121	79	y
26	1A	Substitution	58	3A	:	90	5A	Z	122	7A	z
27	1B	Escape	59	3B	;	91	5B	[	123	7B	{
28	1C	File separator	60	3C	<	92	5C	\	124	7C	|
29	1D	Group separator	61	3D	=	93	5D	]	125	7D	}
30	1E	Record separator	62	3E	>	94	5E	^	126	7E	~
31	1F	Unit separator	63	3F	?	95	5F	_	127	7F	□

Tabel 3.2: Tabel ASCII (didapat dari http://www.cdrummond.qc.ca)

Berdasarkan data dari tabel tersebut, algoritma penulis hanya mengambil representasi desimal mulai dari 33 hingga 126, yaitu semua karakter alfanumerik ditambah karakter simbolik yang berada di keyboard layout standar ANSI-INCITS.

Berikut adalah algoritma untuk menggeser sebuah karakter sebesar sebuah konstanta KEY dalam notasi *pseudocode* yang mirip pascal:

```
procedure EncryptChar (input/output c :
char)
   {Mengubah karakter c menjadi karakter
   baru sesuai dengan penggeseran
   representasi desimalnya dalam ASCII}

KAMUS LOKAL
   i : integer

ALGORITMA
   i <- CharToDecASCII(c)
   i <- 33 + (i + KEY)mod93
   c <- DecToCharASCII(i)
```

Berikut adalah algoritma untuk mengenkripsi sebuah string dalam notasi *pseudocode* yang mirip pascal:

```
procedure Encrypt (input/output S : string)
   {Mengubah string S menjadi string baru
   dengan merubah string tersebut per satu
   karakter}
```

```
KAMUS LOKAL
   i : integer

ALGORITMA
   for (i <- 1) to length(S) do
        EncryptChar(S[i])
```

Algoritma tersebut termasuk salah satu teknik enkripsi yang sangat sederhana dan mudah untuk dipecahkan oleh orang yang memiliki pemahaman yang cukup dalam bidang enkripsi dan kode-kode sederhana, namun ini adalah salah satu algoritma yang menggunakan basis dari teori modulo dengan baik dan mengombinasikannya dengan teori enkoding ASCII.

Berikut adalah contoh penerapan algoritma enkripsi ini pada penyimpanan password di suatu server:

Data sebenarnya			**Data pada server**	
Username	Password		Username	Password
muhr	rumahku		muhr	HKC7>AK
Reza32	IniPaSsw0rd		Reza32	|D?&7I)McH:
1rmdhn2	123abcd		1rmdhn2	def789;
m.reza.r	aSdFqWeRt		m.reza.r	7):yG-;(J

Tabel 3.3: Contoh hasil dari algoritma enkripsi sederhana penulis dengan KEY = 111

III. Analisis Kelebihan dan Kelemahan

A. *Kelebihan*

Kelebihan utama dari proses enkripsi adalah kecepatan algoritma enkripsi itu dilakukan. Sekompleks apapun algoritma enkripsi yang ada ataupun sebesar apapun key yang digunakan dalam algoritma enkripsi semua algoritma enkripsi memiliki kompleksitas O(1) karena tidak ada perubahan kompleksitasnya terhadap berapa besar pun input yang ada.

Kelebihan lainnya yang penulis temukan dari metode penyimpanan password dengan enkripsi ini adalah dengan teknik enkripsi kita dapat merubah kembali data yang disimpan dengan password pengguna. Hal ini berguna dalam kasus dimana pengguna melupakan passwordnya sendiri dan ingin mendapatkan kembali password lamanya. Tapi sebenarnya hal ini juga bisa menjadi kelemahan karena jika ada seseorang yang mendapat akses ke password awal hasil permintaan pengguna dan data hasil enkripsi untuk password itu maka ia bisa mendapatkan fungsi yang kita gunakan untuk mengenkripsi password tersebut.

Pengembalian password yang terlupakan hanya bisa dihasilkan dari metode penyimpanan password dalam plaintext serta metode ini, sedangkan pada metode menggunakan fungsi hash password yang telah disimpan pada server tidak bisa dikembalikan ke dalam bentuk password yang sebenarnya.

B. *Kekurangan*

Secara singkat, dapat dibilang bahwa metode penyimpanan dengan menggunakan enkripsi adalah salah satu metode terburuk yang ada setelah penyimpanan password dalam plaintext, hal ini dapat terlihat bahwa:

1. Khusus algoritma enkripsi dengan “penggeseran” yang penulis gunakan, terlihat dengan jelas bahwa ketika password yg digunakan adalah 123abcd maka hasil enkripsi adalah def789; ini jelas menunjukkan bahwa algoritma enkripsi yang digunakan menggunakan semacam teknik penggeseran, seseorang hanya perlu mencari key yang tepat untuk mendapatkan fungsi enkripsi keseluruhan.
2. Salah satu kekurangan dari enkripsi adalah jika seorang memiliki kemampuan khusus dalam melakukan reverse-engineering pada kode assembly di server yang digunakan untuk enkripsi password, maka ia bisa dengan mudah mendapatkan fungsi enkripsi yang digunakan lalu membuat fungsi dekripsi yang sesuai. Setelah ia mendapatkan fungsi dekripsi dan data password terenkripsi, bukan hal sulit baginya untuk mendapatkan data username dan password untuk seluruh pengguna dalam server itu.
3. Jika suatu algoritma enkripsi memiliki kunci tertentu seperti yang digunakan dalam algoritma penulis maka biasanya key tersebut juga akan tersimpan dalam server yang sama. Jika seseorang berhasil masuk untuk mendapatkan data password terenkripsi maka dengan logika sederhananya ia juga bisa mendapatkan key untuk memecahkan algoritma enkripsi tersebut.
4. Pada sebagian fungsi enkripsi, input sebuah data yang sama pasti akan menghasilkan *ciphertext* yang sama, dengan demikian mudah bagi penyerang untuk membuat sebuah akun baru dengan sandi umum seperti 12345, mengecek *ciphertext* hasil dekripsi password tersebut, lalu mencari *ciphertext* yang sama pada data terenkripsi yang telah ia dapatkan sebelumnya. Dengan demikian ia bisa mendapatkan akses ke akun-akun yang menggunakan *password* umum dengan mudah.

IV. Kesimpulan

Penggunaan metode penyimpanan password dengan menggunakan enkripsi bukan merupakan cara terbaik yang ada. Sebagus dan sekuat apapun enkripsi yang digunakan dalam penyimpanan password, setiap fungsi enkripsi masih rentan dengan fakta bahwa untuk melakukan reverse-engineering pada fungsi enkripsi tersebut mungkin terjadi. Reverse-engineering sebuah fungsi enkripsi memang susah, tapi tidak mustahil untuk dilakukan demi mendapatkan sebuah fungsi dekripsi yang berguna dalam berbagai kemungkinan.

Metode penyimpanan password yang saat ini sering digunakan adalah mengombinasikan fungsi encryption hash dengan terlebih dahulu menambahkan salt – random string dengan panjang tertentu – pada password user sehingga bahkan jika ada dua user dengan password yang

sama dapat menghasilkan hash yang berbeda.

Metode penyimpanan password hingga saat ini masih terus berkembang dan tidak ada satupun metode yang kebal dari berbagai macam tipe serangan. Bahkan metode fungsi hash ditambah salt sendiri masih rentan jika diserang dengan metode bruteforce (mencoba semua kemungkinan karakter password) atau metode dictionary attack (mencoba semua kemungkinan password yang umum digunakan).

VII. Ucapan Terima Kasih

Penulis mengucapkan terima kasih kepada Pak Rinaldi Munir atas kelasnya yang membuat penulis tertarik pada kriptografi, serta menyadarkan penulis bahwa kriptografi tidak selamanya rumit. Terima kasih juga untuk Brady Haran dan Tom Scott dalam videonya di youtube.com yang telah menjadi inspirasi utama penulis dalam membuat paper ini.

Daftar pustaka

[1] Munir, Rinaldi. "Diktat Kuliah IF2120 Matematika Diskrit". Program Studi Teknik Informatika, 2006, Bandung, Indonesia.

[2] What is encryption? http://searchsecurity.techtarget.com/definition/encryption diakses pada 8 Desember 2015

[3] How NOT to Store Password – Computerphile https://youtu.be/8ZtInClXe1Q diakses pada 5 Desember 2015

[4] http://www.darkreading.com/safely-storing-user-passwords-hashing-vs-encrypting/a/d-id/1269374 diakses pada 7 Desember 2015

[5] The Extended ASCII Chart http://www.cdrummond.qc.ca/cegep/informat/professeurs/alain/files/ascii.htm diakses pada 7 Desember 2015

[6] Encryption Generator – Encrypt/Decrypt Text Online http://textmechanic.com/Encryption-Generator.html diakses pada 8 Desember 2015

Pernyataan

Dengan ini penulis menyatakan bahwa makalah yang penulis tulis ini adalah tulisan penulis sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 8 Desember 2015

Muhammad Reza Ramadhan
13514107