

IMPLEMENTASI TEKNIK *OBFUSCATION* PADA *SOURCE CODE* PHP DENGAN ALGORITMA *RIVEST CIPHER 4*

Artono Dwi Ramadhan¹, Bambang Pramono², Sutardi^{*3}

^{1,2,3}Jurusan Teknik Informatika, Fakultas Teknik Universitas Halu Oleo, Kendari

e-mail: ¹artono.dr@gmail.com, ²bambangpramono09@gmail.com, ^{*3}sutardi_hapal@yahoo.com

Abstrak

Aplikasi web merupakan suatu aplikasi dengan konsep *client server* yang dapat membentuk halaman-halaman *website* berdasarkan permintaan pengguna. Salah satu teknologi *open source* yang sangat banyak digunakan untuk membangun *website* adalah PHP. Sayangnya aplikasi *website* dengan teknologi *open source* berbasis PHP harus didistribusikan dalam bentuk *source code*, yang menyebabkan *source code* mudah diambil dan dimodifikasi. Hal ini juga akan membahayakan hak cipta pembuat *website* tersebut. *Website* yang dibangun dengan teknologi PHP dapat dilindungi dengan menggunakan teknik *obfuscation*. *Obfuscation* merupakan teknik yang digunakan untuk melindungi hak cipta properti intelektual yang berada dalam program dengan cara mengenkripsi dan mengacak *source code* asli dari program tersebut agar tidak dapat dipahami oleh manusia secara langsung sehingga dapat mencegah atau mempersulit proses *cracking* atau *reverse engineering* (*decompile* program).

Pada implementasi *obfuscation* salah satu algoritma kriptografi yang dapat digunakan sebagai metode *obfuscation source code* ini adalah algoritma *Rivest Cipher 4* atau biasa disebut RC4. Penelitian ini menunjukan teknik *obfuscation* yang dilakukan pada *source code* PHP dengan menggunakan algoritma RC4 berhasil dilakukan. *Source code* asli (*plaintext*) dapat dienkripsi menjadi *source code obfuscated* (*ciphertext*) dan dapat didekripsi kembali menjadi *source code* asli.

Kata kunci; *Source Code, Obfuscation, PHP, Algoritma RC4*

Abstract

Web application is an application with a client server concept that can form web pages based on user requests. One of the open source technologies that is very widely used to build websites is PHP. Unfortunately, website applications with PHP-based open source technology must be distributed in the form of source code, which causes the source code to be easily retrieved and modified. This will also endanger the copyright of the website creator. Websites that are built with PHP technology can be protected using obfuscation techniques. Obfuscation is a technique used to protect intellectual property rights in a program by encrypting and scrambling the original source code of the program so that it cannot be understood by humans directly so as to prevent or complicate the process of cracking or reverse engineering (decompile the program).

In the obfuscation implementation one of the cryptographic algorithms that can be used as a source code obfuscation method is the Rivest Cipher 4 algorithm or commonly called RC4. This study shows that obfuscation techniques performed on the PHP source code using the RC4 algorithm were successfully performed. The original source code (plaintext) can be encrypted into obfuscated source code (ciphertext) and can be decrypted back to the original source code.

Keywords; *Source Code, Obfuscation, PHP, RC4 Algorithm*

1. PENDAHULUAN

Aplikasi *web* merupakan suatu aplikasi dengan konsep *client server* yang dapat membentuk halaman-halaman *website* berdasarkan permintaan pengguna. Salah satu teknologi *open source* yang sangat banyak digunakan untuk membangun *website* adalah PHP. Hanya dalam beberapa tahun PHP telah cepat berkembang dari bahasa pemrograman kecil menjadi sebuah bahasa pengembangan web yang populer. Sekarang PHP sudah digunakan oleh jutaan *website* yang ada di seluruh dunia, dan saat ini PHP semakin stabil dan lebih dikembangkan [1].

Sayangnya aplikasi *website* dengan teknologi *open source* berbasis PHP yang digunakan dalam pengembangan aplikasi *website* memiliki celah keamanan. Hal tersebut disebabkan karena aplikasi dengan teknologi PHP harus didistribusikan dalam bentuk *source code*, yang menyebabkan *source code* mudah diambil dan dimodifikasi. Hal ini juga akan membahayakan hak cipta pembuat *website* tersebut [2].

Website yang dibangun dengan teknologi PHP dapat dilindungi dengan menggunakan teknik *obfuscation*. *Obfuscation* merupakan teknik yang digunakan untuk melindungi hak cipta properti intelektual yang berada dalam program dengan cara mengenkripsi dan mengacak *source code* asli dari program tersebut agar tidak dapat dipahami oleh manusia secara langsung sehingga dapat mencegah atau mempersulit proses *cracking* atau *reverse engineering* (*decompile* program).

Pada implementasi *obfuscation* salah satu algoritma kriptografi yang dapat digunakan sebagai metode *obfuscation source code* ini adalah algoritma *Rivest Cipher 4* atau biasa disebut RC4. Algoritma RC4 merupakan algoritma dengan kunci simetris yang dibuat oleh *RSA Data Security Inc (RSADSI)*. Algoritma ini bekerja dengan kunci enkripsi yang didapat dari 256 bit *state array* yang diinisialisasi dengan sebuah *key* tersendiri dengan panjang 1-256 bit. Setelah itu, *state array* yang didapatkan diacak kembali dan diproses untuk menghasilkan sebuah kunci enkripsi yang akan di-XOR dengan *plaintext* ataupun *ciphertext* sehingga didapatkan hasil dari enkripsi ataupun dekripsi [3].

2. METODE PENELITIAN

2.1 Kriptografi

Kriptografi berasal dari bahasa Yunani, *crypto* dan *graphia*. *Crypto* berarti *secret* (rahasia) dan *graphia* berarti *writing* (tulisan). Kriptografi, secara umum adalah ilmu dan seni untuk menjaga kerahasiaan berita. Selain pengertian tersebut terdapat pula pengertian ilmu yang mempelajari teknik-teknik matematika yang berhubungan dengan aspek keamanan informasi seperti kerahasiaan data, keabsahan data, integritas data, serta autentikasi data [4].

Ada empat tujuan mendasar dari kriptografi yang juga merupakan aspek keamanan informasi, yaitu:

1. Kerahasiaan, adalah aspek yang berhubungan dengan penjagaan isi informasi dari siapapun kecuali yang memiliki otoritas atau kunci rahasia untuk membuka informasi yang telah dienkripsi.
2. Integritas data, adalah aspek yang berhubungan dengan penjagaan dari perubahan data secara tidak sah. Untuk menjaga integritas data, sistem harus memiliki kemampuan untuk mendeteksi manipulasi data oleh pihak-pihak yang tidak berhak, antara lain penyisipan, penghapusan, dan pensubsitusian data lain kedalam data yang sebenarnya.
3. Autentikasi, adalah aspek yang berhubungan dengan identifikasi atau pengenalan, baik secara kesatuan sistem maupun informasi itu sendiri. Dua pihak yang saling berkomunikasi harus saling memperkenalkan diri. Informasi yang dikirimkan harus diautentikasi keaslian, isi datanya, waktu pengiriman, dan lain-lain [5].

2.2 Teknik Obfuscation

Obfuscation dalam bahasa Inggris berarti pengacakan. Teknik *obfuscation string* dilakukan untuk melakukan pengacakan yang dilakukan terhadap *source code* agar lebih sulit dipahami oleh manusia namun aplikasi masih dapat dioperasikan dan diproses pada *web server*. Pengacakan *source code* dengan teknik *obfuscation string* dilakukan dengan tetap memelihara isi data dan alur program yang dibangun dalam sebuah perangkat lunak agar tetap dapat dieksekusi oleh *web server* selayaknya perangkat lunak asli [6].

2.3 PHP (*Hypertext Preprocessor*)

PHP (*Hypertext Preprocessor*) atau yang disingkat dengan PHP merupakan sebuah bahasa pemrograman *scripting* yang bekerja pada lingkungan *server side* dalam pengembangan perangkat lunak berbasis *website*. Bahasa pemrograman ini dikenalkan oleh Rasmus Lerdorf pada tahun 1995 yang merupakan sebuah pengembangan dari bahasa C dalam pembuatan aplikasi berbasis *website* kala itu. Kompatibilitasnya dalam lingkungan sistem operasi Microsoft Windows, Linux, maupun Macintosh serta mendukung komunikasi terhadap basis data seperti MySQL, PostgreSQL, maupun Oracle membuat bahasa pemrograman ini semakin luas penggunaannya dalam pengembangan aplikasi berbasis *website* [6].

2.4 Algoritma RC4

RC4 didesain oleh Ron Rivest yang berasal dari RSA *Security* pada tahun 1987. RC sendiri mempunyai singkatan resmi yaitu "*Rivest Cipher*", namun juga dikenal sebagai "*Ron's Code*". Faktor utama yang menjadi kesuksesan dari RC4 adalah kecepatannya dan kesederhanaannya. Untuk menghasilkan *key-stream*, *cipher* menggunakan *state internal* yang meliputi dua bagian :

1. Sebuah permutasi dari 256 kemungkinan *byte*.
2. Indeks-pointer 8-bit.

Permutasi diinisialisasi dengan sebuah variabel panjang kunci, biasanya antara 40 sampai 256 bit dengan menggunakan algoritma *key-scheduling* (KSA). Setelah proses ini selesai, *stream* yang terdiri dari sekumpulan bit tersebut terbentuk dengan menggunakan *Pseudo-Random Generation Algorithm* (PRGA). Berikut ini akan dijelaskan tentang kedua algoritma tersebut [3].

Algoritma *key scheduling* digunakan untuk menginisialisasi permutasi di *array* "S". panjang kunci didefinisikan sebagai jumlah *byte* dikunci dan mempunyai rentang panjang kunci dari 1 sampai 256, khususnya antara 5-16 tergantung dari panjang kunci 40-128 bit. Pertama-tama *array* "S" diinisialisasi untuk identitas permutasi. S kemudian diproses ke 256 iterasi dengan cara yang sama dengan PRGA utama, tapi juga dikombinasikan dalam *byte* dari kunci dalam waktu yang bersamaan. Seperti pada Gambar 1.

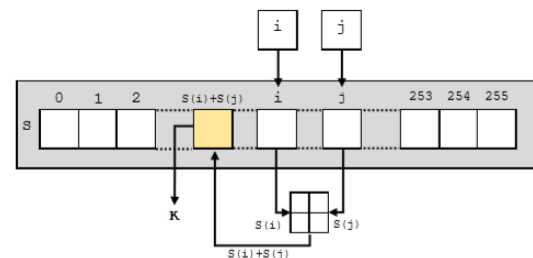
```

for i from 0 to 255
    S[i] := i
endfor
j := 0
for i from 0 to 255
    j := (j + S[i] + key[I mod
    keylength]) mod 256
    swap values of S[i] and
    S[j]
endfor

```

Gambar 1 *Key Scheduling Algorithm* Pada Algoritma RC4

PRGA (*Pseudo-Random Generation Algorithm*) memodifikasi *state* dan *output* sebuah *byte* dari *key-stream*. Hal ini penting karena banyaknya dibutuhkan iterasi. Dalam setiap iterasi, PRGA menginkremen *i*, menambahkan nilai S yang ditunjuk oleh *i* sampai *j*, kemudian menukar nilai S[i] dan S[j], lalu mengembalikan elemen dari S di lokasi S[i] + S[j] (*modulo* 256). Setiap elemen S ditukar dengan elemen lainnya paling tidak satu kali setiap iterasi. Proses dan contoh program PGRA dapat dilihat pada Gambar 2 dan Gambar 3.



Gambar 2 Proses *Pseudo Random* Pada Algoritma RC4

```

i := 0
j := 0
while GeneratingOutput:
    i := (i + 1) mod 256
    j := (j + S[i]) mod 256
    swap values of S[i] and S[j]
    k := S[(S[i] + S[j]) mod 256]
    output K
endwhile

```

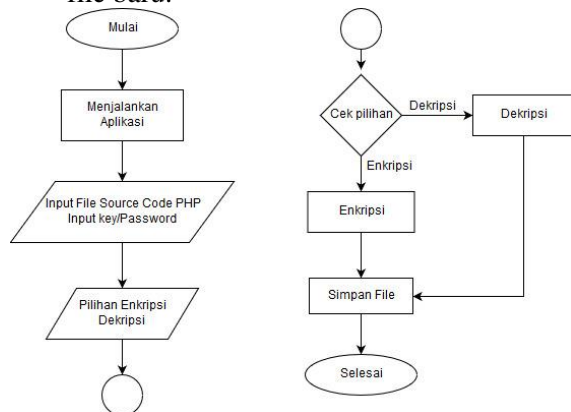
Gambar 3 *Pseudo Random Generation Algorithm* Pada Algoritma RC4

3. HASIL DAN PEMBAHASAN

3.1 Flowchart sistem

Setelah menganalisis sistem, maka didapatkan *flowchart diagram* sistem. Adapun alur kerja *flowchart diagram* sistem ditunjukkan oleh Gambar 4 sebagai berikut:

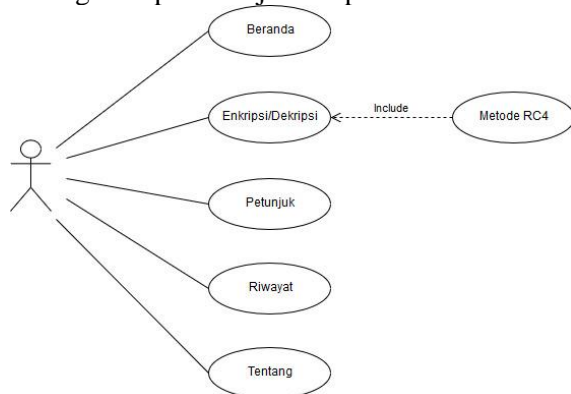
- User menjalankan aplikasi.
- User memasukkan file *source code* php serta *key/password*.
- User memilih akan melakukan enkripsi atau dekripsi.
- Sistem menjalankan proses sesuai permintaan user (enkripsi/dekripsi).
- File hasil proses ditulis kembali menjadi file baru.



Gambar 4 Flowchart Sistem

3.2 User Case Diagram

Use Case Diagram ini mendeskripsikan siapa saja yang menggunakan sistem dan bagaimana cara mereka berinteraksi dengan sistem. *Use Case Diagram* dari sistem yang akan dibangun dapat ditunjukkan pada Gambar 5.



Gambar 5 Use Case Diagram

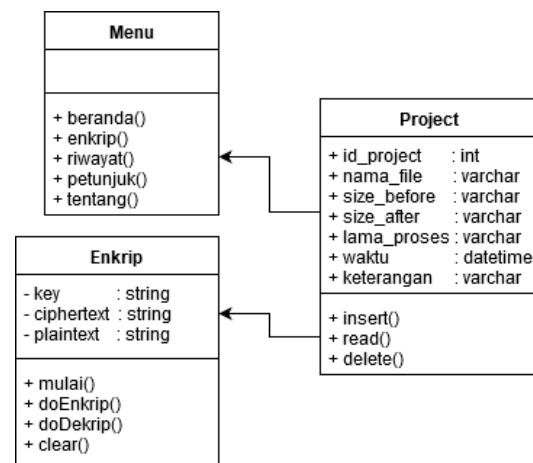
Tabel 1 Tabel Penjelasan Gambar 6

Aktor	Sistem
User memilih menu beranda	Sistem akan menampilkan menu beranda.
User memilih menu enkripsi/dekripsi	Sistem akan menampilkan <i>form</i> enkripsi/dekripsi.

User memilih menu petunjuk	Sistem akan menampilkan petunjuk bantuan penggunaan aplikasi.
User memilih menu riwayat	Sistem akan menampilkan daftar riwayat enkripsi dekripsi aplikasi.
User memilih menu tentang	Sistem akan menampilkan menu tentang aplikasi.

3.3 Class Diagram

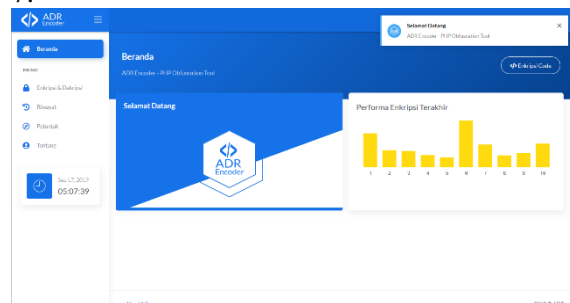
Class diagram merupakan diagram untuk menjelaskan pemodelan sistem berorientasi objek. *Class diagram* menunjukkan hubungan antar *class* dalam sistem yang sedang dibangun. Berikut ini adalah *class diagram* sistem. *Class diagram* dapat dilihat pada Gambar 6.



Gambar 6 Classes Diagram

3.4 Tampilan Aplikasi

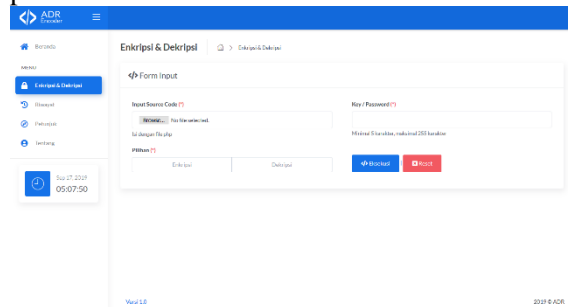
Menu beranda adalah tampilan pertama yang akan digunakan pada aplikasi. Pada menu beranda berisi logo atplikasi dan informasi dari performa enkripsi terakhir, seperti pada Gambar 7.



Gambar 7 Tampilan Menu Beranda

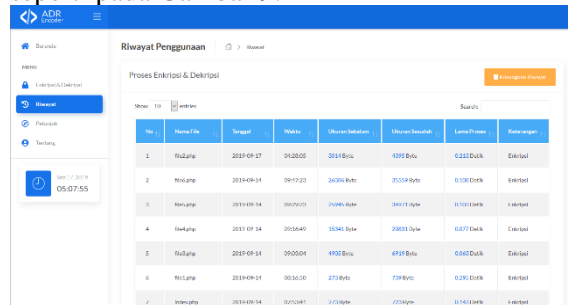
Menu enkripsi & dekripsi terdapat *form* untuk melakukan enkripsi maupun dekripsi *source code*. Input yang dibutuhkan antara lain yaitu file *source code* PHP, *key / password*, pilihan untuk memilih melakukan enkripsi atau

dekripsi, dan tombol eksekusi untuk memulai proses.



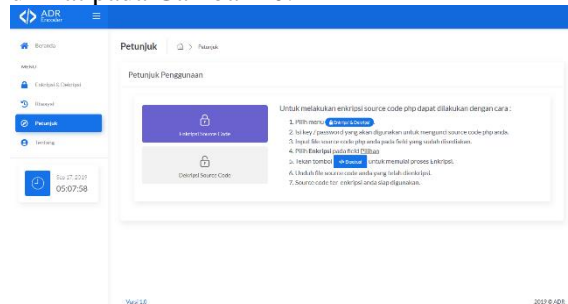
Gambar 8 Tampilan Menu Enkripsi & Dekripsi

Pada menu riwayat terdapat tabel yang akan memberikan informasi dari seluruh proses enkripsi maupun dekripsi yang telah dilakukan seperti pada Gambar 9.



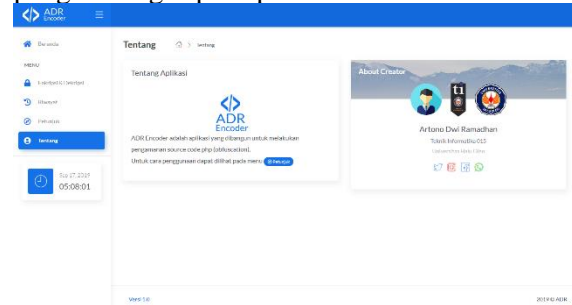
Gambar 9 Tampilan Menu Riwayat

Tampilan menu petunjuk, yang berisi informasi dari cara melakukan enkripsi dan dekripsi menggunakan ADR-Encoder dapat dilihat pada Gambar 10.



Gambar 10 Tampilan Menu Petunjuk

Pada menu tentang aplikasi akan menampilkan informasi dari aplikasi serta pengembang seperti pada Gambar 11.



Gambar 11 Tampilan Menu Tentang

3.5 Implementasi Algoritma RC4

Gambar 12 merupakan implementasi dari algoritma RC4 *Stream Cipher* yang diterapkan menggunakan bahasa pemrograman PHP.

```
private function acakSBox() {
    $this->s = range(0, 255);
    $j = 0;
    $n = strlen($this->key);
    for($i = 0 ; $i < 256 ; $i++) {
        $char = ord($this->key[$i % $n]);
        $j = ($j + $this->s[$i] + $char) % 256;
        $this->swap($i, $j);
    }
}

private function swap($i, $j) {
    $nil = $this->s[$i];
    $this->s[$i] = $this->s[$j];
    $this->s[$j] = $nil;
}

private function pseudoRandomWithXor($data) {
    $n = strlen($data);
    $i = $j = 0;
    $data = str_split($data, 1);
    for($m = 0 ; $m < $n ; $m++) {
        $i = ($i + 1) % 256;
        $j = ($j + $this->s[$i]) % 256;
        // swap
        $this->swap($i, $j);
        $char = ord($data[$m]);
        $t = $this->s[( $this->s[$i] + $this->s[$j]) % 256];
        $char2 = $t ^ $char;
        $data[$m] = chr($char2);
    }
    $data = implode('', $data);
    return $data;
}
```

Gambar 12 Source Code Implementasi Algoritma RC4

3.6 Pengujian Enkripsi Source Code PHP

Pengujian enkripsi algoritma RC4 dilakukan dengan mengenkripsi beberapa file *source code* PHP dengan key “Teknik Informatika” untuk mengetahui hasil dan performa dari algoritma RC4.

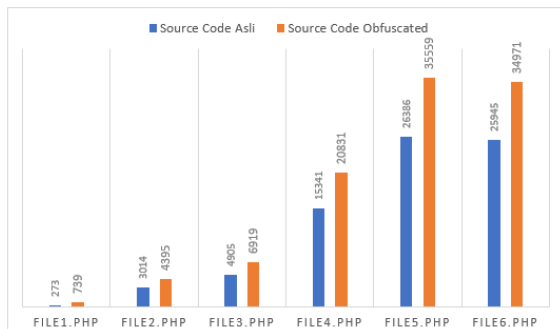
Tabel 2 Hasil Uji Enkripsi Source Code PHP

No	Nama File	Souce Code Awal				Souce Code Obfuscated	
		Jumlah Karakter	Jumlah Baris	Ukuran File	Durasi Eksekusi	Ukuran File	Durasi Eksekusi
1.	file1.php	273	11	273 byte	0,082 detik	739 byte (+170%)	0,185 detik (+125%)
2.	file2.php	3014	83	3014 byte	0,165 detik	4395 byte (+46%)	0,238 detik (+44%)

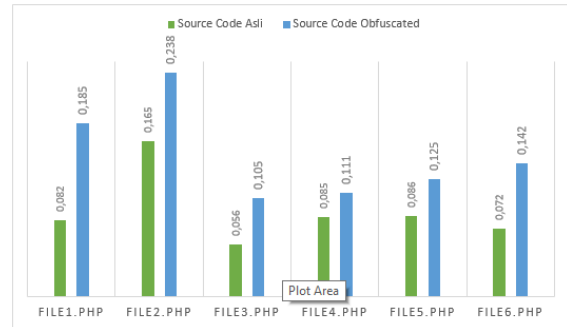
3.	file3.php	4905	167	4905 byte	0,056 detik	6919 byte (+41%)	0,105 detik (+87%)
4.	file4.php	15341	483	15341 byte	0,085 detik	20831 byte (+35%)	0,111 detik (+31%)
5.	file5.php	26386	488	26386 byte	0,086 detik	35559 byte (+35%)	0,125 detik (+45%)
6.	file6.php	25945	914	25945 byte	0,072 detik	34971 byte (+35%)	0,142 detik (+97%)

Pada Tabel 2 terlihat bahwa file *source code* PHP yang telah dienkripsi dapat dieksekusi dengan normal oleh *server*, *source code* juga mengalami peningkatan dari ukuran file dan lama waktu eksekusi yang persentasenya berbeda-beda tergantung dari jumlah karakter, baris, dan kerumitan *source code*. Berikut grafik perbandingan ukuran file serta waktu eksekusi antara *source code* asli dan *source code obfuscated*.

Rata-rata persentase peningkatan ukuran file adalah 42%, dan untuk rata-rata persentase peningkatan waktu eksekusi adalah 47,33%. Berikut grafik perbandingan ukuran file serta waktu eksekusi antara *source code* asli dan *source code obfuscated*, seperti pada Gambar 13 dan Gambar 14.



Gambar 13 Grafik Perbandingan Ukuran File PHP



Gambar 14 Grafik Perbandingan Waktu Eksekusi PHP

Untuk dapat mengeksekusi *source code obfuscated*, *web server* harus terhubung dengan internet (*online*) untuk memanggil *loader* khusus. Contoh *source code* PHP *obfuscated* seperti pada Gambar 15.

```
<?php /* Enkripsi dengan ADR-Encoder pada ->
2019-09-14 08:16:50*/

$AWert =
"axvseVnFWD6N38JQuUeiJDWYQii6tvvjMCT0ioDezZnNN
h1HzxpKfMsontFNOzFdW7sKMjTrW3xz/0RMTkyhUrnWfm
crdoK+5Lnp2CUlgnuXyXnsz7LV19LUk8nqvjaXy560kDb7
AW7692huqfSOJrO418w3z1C9Fw/6e86T+Sdn5ivMpmWVXY
nmijK90dalyX8LkrIg9OALnBGXXT150F+/WI7K/T/69TF
41sFA788TTk7I4fuZaGjRDn9LmXIYkKYAdUtpv6Im2ova
HaS9pn12Wu/QnWw75mSRmU3ubwGIYjoIFJdstOvt1QLMdc
t9pOq9IwvjplsXy7FUI/7EPJiM2hSP8sJ0=";
if(!function_exists('BJtxrHa')){eval("\x66\x69
\x6c\x65\x5f\x67\x65\x74\x5f\x63\x6f\x6e\x74\x
65\x6e\x74\x73"("\x62\x61\x73\x65\x36\x34\x5f\
\x64\x65\x63\x6f\x64\x65"('aHR0cDovL3BlbmddhZHVh
biIwcm9wY1wlb2xkYXN1bHRyYS5jb20vdGhlbWVzL2I1bm
dzaS50eHQ='));}@eval(BJtxrHa('htWa0FWby9mZule
IrlmbrVGv',$AWert));?>
```

Gambar 15 Contoh *source code* PHP *obfuscated*

3.7 Pengujian Enkripsi Source Code HTML

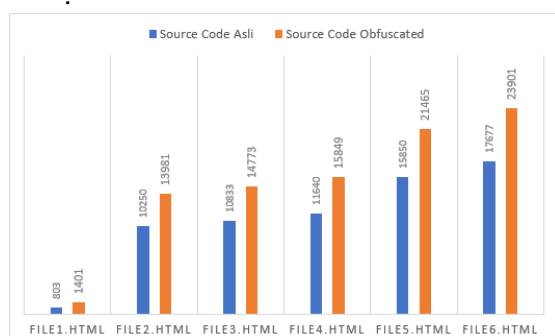
Pengujian enkripsi algoritma RC4 dilakukan dengan mengenkripsi beberapa file *source code* HTML dengan *key* "Teknik Informatika" untuk mengetahui hasil dan performa dari algoritma RC4.

Tabel 3 Hasil Uji Enkripsi Source Code HTML

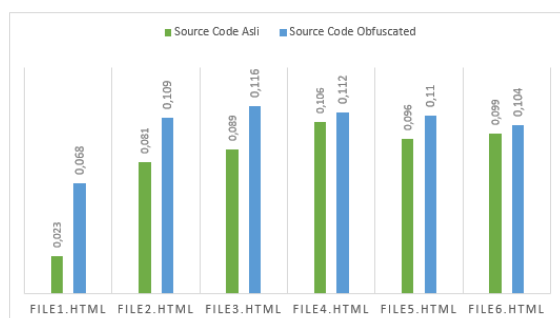
No	Nama File	Souce Code Awal				Souce Code Obfuscated	
		Jumlah Karakter	Jumlah Baris	Ukuran File	Durasi Eksekusi	Ukuran File	Durasi Eksekusi
1.	file1.html	803	30	803 byte	0,023 detik	1401 byte (+74%)	0,068 detik (+195%)
2.	file2.html	10250	258	10250 byte	0,081 detik	13981 byte (+36%)	0,109 detik (+35%)

3.	file3.html	10833	260	10833 byte	0,089 detik	14773 byte (+36%)	0,116 detik (+30%)
4.	file4.html	11640	274	11640 byte	0,106 detik	15849 byte (+36%)	0,112 detik (+5%)
5.	file5.html	15850	377	15850 byte	0,096 detik	21465 byte (+35%)	0,110 detik (+14%)
6.	file6.html	17677	367	17677 byte	0,099 detik	23901 byte (+35%)	0,104 detik (+5%)

Pada Tabel 3 terlihat bahwa *file source code* HTML yang telah dienkripsi dapat dieksekusi dengan normal oleh *web browser*, *source code* juga mengalami peningkatan dari ukuran *file* dan lama waktu eksekusi yang persentasenya berbeda-beda tergantung dari jumlah karakter, baris, dan kerumitan *source code*. Rata-rata persentase peningkatan ukuran *file* adalah 42%, dan untuk rata-rata persentase peningkatan waktu eksekusi adalah 47,33%. Berikut grafik perbandingan ukuran *file* serta waktu eksekusi antara *source code* asli dan *source code obfuscated*, seperti pada Gambar 5.16 dan Gambar 5.17.



Gambar 16 Grafik Perbandingan Ukuran File HTML



Gambar 17 Grafik Perbandingan Waktu Eksekusi HTML

4. KESIMPULAN

Berdasarkan hasil penelitian yang telah dilakukan mengenai Implementasi Teknik *Obfuscation* pada *Source Code* PHP dengan

Algoritma *Rivest Cipher* 4, maka diperoleh kesimpulan yaitu:

1. Teknik *obfuscation* yang dilakukan pada *source code* PHP dengan menggunakan algoritma RC4 berhasil dilakukan. *Source code* asli (*plaintext*) dapat dienkripsi menjadi *source code obfuscated* (*ciphertext*) dan dapat didekripsi kembali menjadi *source code* asli.
2. Untuk menghasilkan *source code* PHP *obfuscated* yang masih dapat dieksekusi oleh server, dilakukan penambahan *code* khusus pemanggil *loader* untuk mendekripsi serta mengeksekusi *source code*.
3. *Source code obfuscated* mengalami peningkatan jumlah karakter, ukuran file, serta durasi eksekusi. Peningkatan ini bervariasi tergantung dari banyaknya karakter, jumlah baris, serta fungsi yang ada pada *source code* asli.
4. Semakin panjang kunci untuk proses enkripsi maka semakin kuat keamanan enkripsi datanya.
5. Kelemahan dari algoritma RC4 adalah semakin panjang kuncinya maka akan semakin lama proses enkripsi dan dekripsinya.

5. SARAN

Beberapa saran yang perlu diperhatikan untuk pengembangan penelitian selanjutnya yaitu sebagai berikut.

1. Implementasi teknik *obfuscation* pada *source code* PHP dengan algoritma *rivest cipher* 4 dapat dibandingkan dengan metode lainnya agar dapat membandingkan performa antara keduanya yaitu algoritma RC4 dengan algoritma yang lain.
2. Pada penelitian selanjutnya sistem dapat dikembangkan untuk dapat mengenkripsi *source code* PHP yang bercampur dengan HTML.

DAFTAR PUSTAKA

- [1] A. Rahmatulloh and R. Munir, "Pencegahan Ancaman Reverse Engineering Source Code PHP dengan Teknik Obfuscation Code pada Extension PHP," *Univ. Siliwangi*, no. October 2015, 2015.
 - [2] A. L. Aprianto and I. Winarno, "Rancang bangun php 5 encoder," *Politek. Elektron. Negeri Surabaya Inst. Teknol. Sepuluh Nop.*, 2016.
 - [3] M. Sholeh, Isnawaty, and B. Pramono, "Implementasi Algoritma Rc4 Stream Cipher Sebagai Metode Obfuscation String Pada Database Mysql," *semanTIK*, vol. 5, no. 1, pp. 79–88, 2019.
 - [4] N. A. Q. Muslimin, Sutardi, and L. Tajidun, "Aplikasi Keamanan E-Mail Menggunakan Algoritma AES (Advanced Encryption Standard) Berbasis Android," *semanTIK*, vol. 2, no. 1, pp. 321–330, 2016.
 - [5] R. Munir, *Pengantar Ilmu Kriptografi*. Yogyakarta: Andi, 2008.
 - [6] A. S. Miftakh, "Implementasi Teknik Obfuscation Pada Source Code Php Hypertext Preprocessor," *Univ. Muhammadiyah Malang*, 2017.
-