ISBN: 978-979-796-238-6

APLIKASI OBFUSCATION UNTUK KEAMANAN KODE SUMBER PADA BAHASA PEMROGAMAN PHP

Maskur[1], Krisma Pradana Putra[2]
[1,2] Teknik Informatika Universitas Muhammadiyah Malang

Kontak Person:
Maskur
Kantor Jurusan Teknik Informatika UMM
GKB 3 lantai 2 Raya Tlogomas 246 Malang, 65144
Telp: 085646554437, E-mail: maskur.informatika@gmail.com

Abstrak

Website merupakan suatu aplikasi yang berjalan dengan menggunakan web server sehingga tidak perlu di *compile* untuk menjalankannya. Ketika kode PHP didistribusikan dalam bentuk *source* dan diunggah ke *server hosting*, maka akan menjadi sangat rentan dan memiliki celah keamanan. Penelitian ini bertujuan untuk menghasilkan aplikasi PHP *encoder* dan *decoder* untuk melindungi hak cipta *source code* PHP dari kegiatan *plagiarism*. Melalui penerapan algoritma kriptografi base64 yang ditambahkan dengan kunci keamanan mengubah *source code* PHP menjadi bentuk cipherteks yang sulit untuk dibaca, tetapi masih dapat berjalan sebagaimana mestinya *source code* PHP asli (plainteks). Aplikasi dibangun menggunakan bahasa pemrograman PHP untuk obfuskasi *source code* PHP dan obfuskasi *file project* PHP. Pengujian dilakukan meliputi obfuskasi yang dilakukan terhadap kode PHP maupun *file project* berupa PHP *Procedural* maupun *Object Oriented Programming*. Dari pengujian yang telah dilakukan masih belum dapat melakukan proses obfuskasi *file project* berupa CMS Wordpress. Pesan *Error* muncul dan terjadi ketika hasil dari obfuskasi dijalankan di *web browser* yang merujuk pada satu file yang mengalami *error* yaitu version.php. Tetapi ketika file tersebut diubah kedalam bentuk plainteks, maka aplikasi dapat berjalan tanpa adanya *error*. Aplikasi ini mampu mengubah *source code* PHP yang sudah dilakukan proses obfuskasi pada website berjalan dengan baik sehingga dapat diterapkan untuk untuk berbagai jenis website. Diharapkan dari penelitian ini mampu mengamankan *source code* dari tindakan yang tidak diinginkan.

Kata kunci : website, obfuskasi, kunci keamanan, *source code*, web server

Pendahuluan

Ketika kode PHP didistribusikan dalam bentuk *source* dan diunggah ke *server hosting*, maka akan menjadi sangat rentan dan memiliki celah keamanan. Kode PHP akan mudah untuk disalin, diubah ataupun digunakan pada aplikasi lain dengan sengaja. Digunakan secara ilegal tanpa ijin dari pemilik kode sumber yang digunakan. Sehingga melindungi kode PHP dari kegiatan *plagiarism* menjadi isu penting atas pesatnya perkembangan industri *web* dinamis dalam perlindungan hak cipta ©copyright [5]. Maka dari permasalahan tersebut dapat dilakukan proses obfuskasi agar *source code* PHP menjadi sulit dibaca dan terhindar dari tindakan *plagiarism*. Teknik obfuskasi umumnya mengubah sintaks *script* tanpa mengubah semantiknya, sehingga walaupun tulisan *script* menjadi susah untuk dibaca namun ketika dieksekusi masih tetap dapat dijalankan seperti sebelum di obfuskasi [4]. Pada penelitian sebelumnya menyampaikan bahwa pembuatan aplikasi *encoder* menggunakan algoritma base64 dan perubahan susunan pada tabel index base64nya [1]. Tentu hal ini masih belum cukup untuk mengamankan *source code* PHP. Kemudian pada penelitian yang lain, aplikasi *encoder* dibuat menggunakan metode *Blowfish* dan penggunaan *file loader* yang ditempatkan di dalam *server* untuk men-*decode* kode PHP yang telah terenkripsi dan menjalankannya [2]. Meskipun metode ini dianggap cukup baik, akan tetapi file dari hasil *encode* memiliki ukuran dua kali lipat lebih besar dari file semula sehingga akan sangat berpengaruh terhadap performa dari *website* dan tidak semua *server hosting* memiliki layanan untuk menjalankan *file loader* yang dibutuhkan untuk proses *decoding* sehingga dirasa tidak cukup *friendly*.

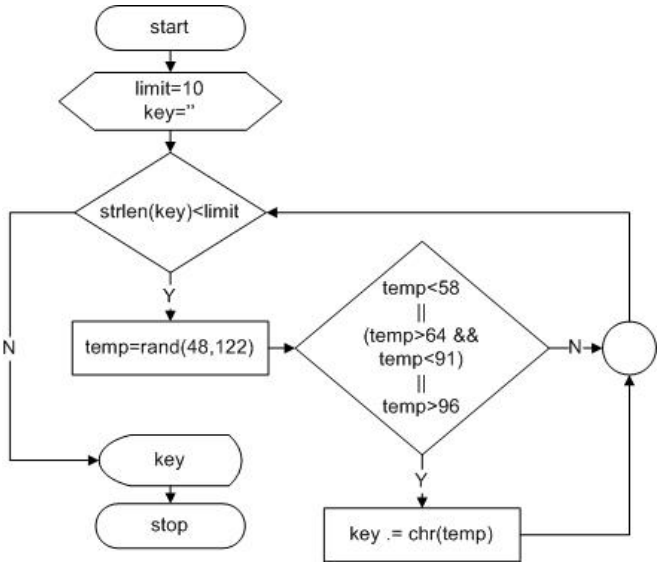
ISBN: 978-979-796-238-6

Berdasarkan dari analisa hasil penelitian yang telah dibahas di atas, maka dapat dilakukan pengembangan yaitu dengan pembuatan aplikasi PHP *encoder* dan *decoder* yang menggunakan algoritma base64 (*friendly* di semua *server hosting* karena merupakan fungsi bawaan PHP) dengan penambahan kunci keamanan.

Metode Penelitian

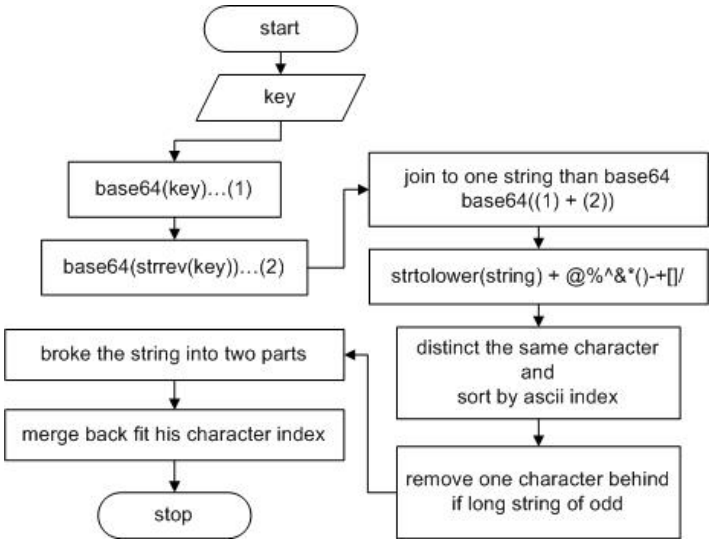
Beberapa tahapan dalam penelitian ini dimulai dari proses pembuatan kunci keamanan sampai melakukan obfuskasi terhadap *source code* website dengan bahasa pemrograman web PHP.

1. Pengolahan Kunci Keamanan
   Referensi kunci dapat diperoleh melalui dua alternatif yaitu inputan dari pengguna atau *key generator*. *Flowchart* algoritma dari *key generator* dapat dilihat pada gambar 1 dibawah ini.



Gambar 1. *Flowchart Key Generator*

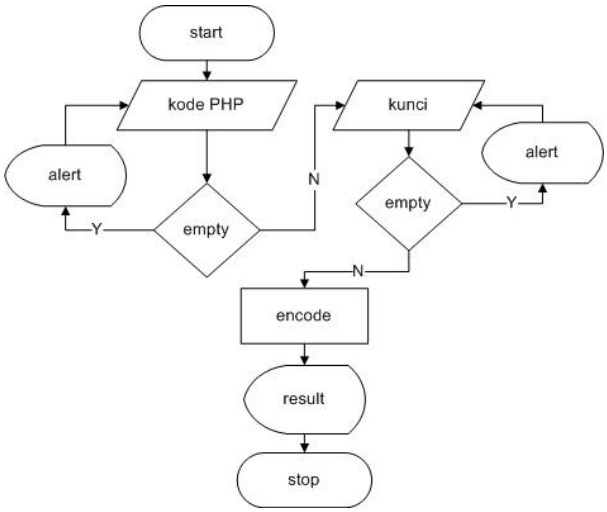
Kemudian kunci akan dilakukan proses enkripsi agar tidak mudah diketahui dan ditebak ketika diinjeksikan kedalam kode PHP hasil obfuskasi. *Flowchart* algoritma enkripsi kunci dapat dilihat pada Gambar 2 dibawah ini.



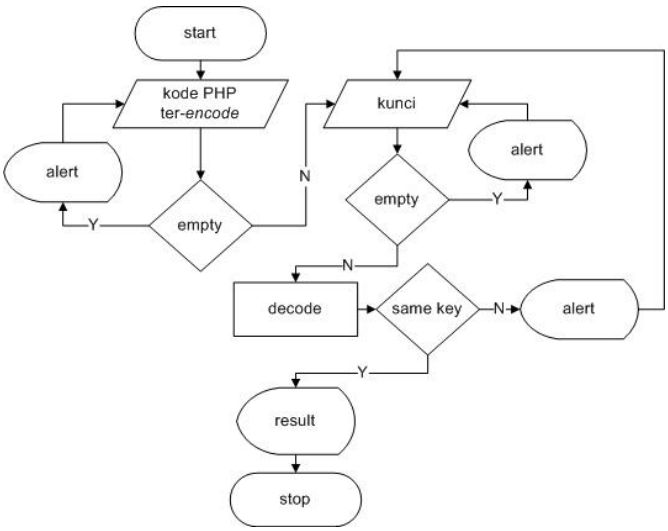
Gambar 2. *Key Encryption*

ISBN: 978-979-796-238-6

2. Proses *Encode* dan *Decode Source Code*
   Proses encode terbagi menjadi dua alternatif yang dapat digunakan untuk melakukan obfuskasi.



Gambar 3. *Flowchart Encode* Kode PHP



Gambar 4. *Flowchart Decode* Kode PHP

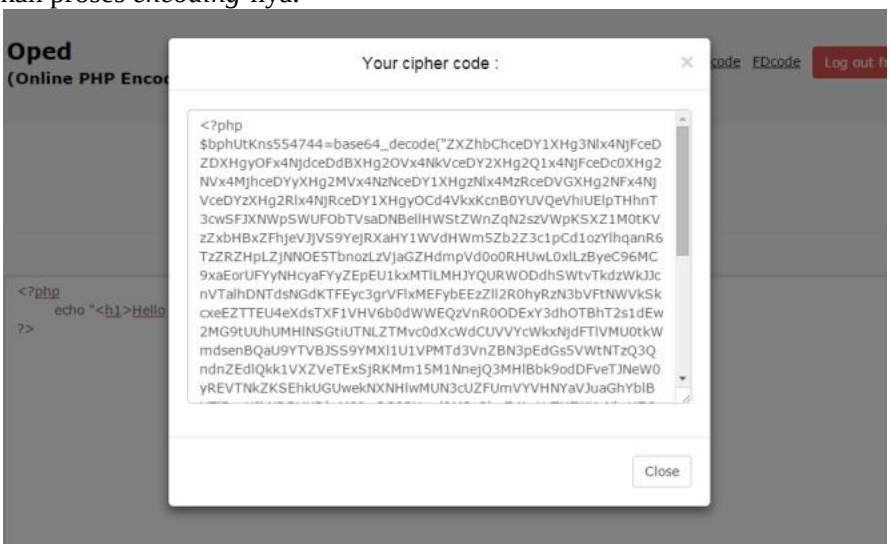
Hasil Penelitian dan Pembahasan

Pengujian sistem dilakukan dengan melakukan proses *encoding* dan *decoding* kode PHP dan *file project* serta melihat hasil dari proses tersebut.

Gambar 5 Percobaan *Encode Script*

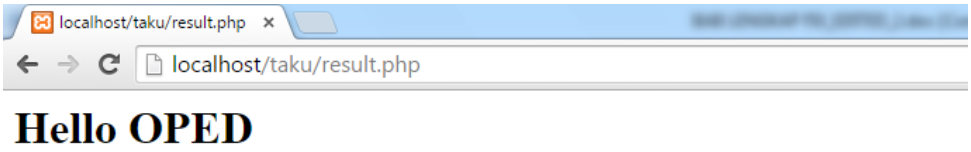
ISBN: 978-979-796-238-6

Setelah memasukkan kode PHP dan kunci keamanan Gambar 5 kemudian tekan tombol *Encode* untuk melakukan proses *encoding*-nya.



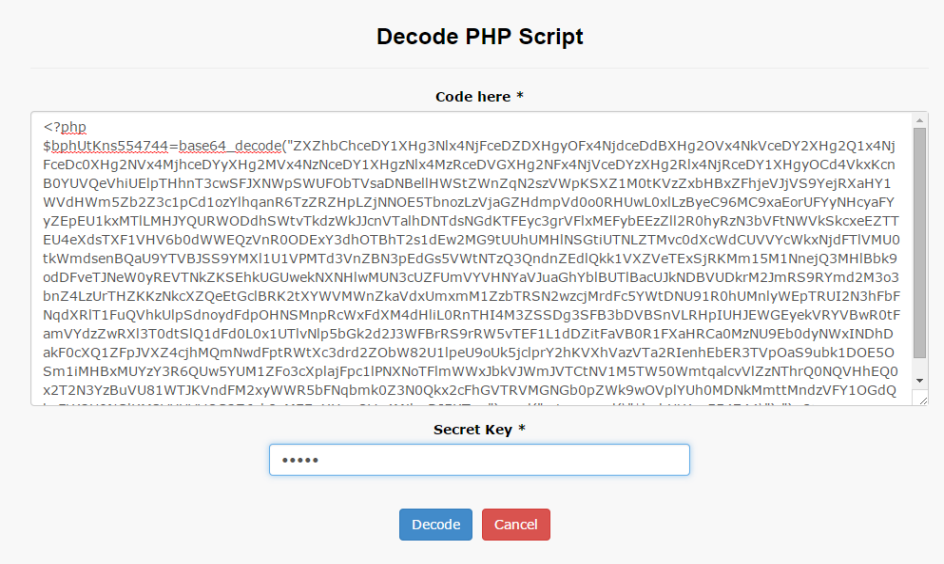
Gambar 6**.** Hasil *Encode Script*

Hasil dari proses *encode* terlihat pada gambar 6 dengan kode PHP yang sudah dalam keadaan acak dan tidak dapat dibaca. Jika kode PHP tersebut dijalankan maka hasilnya akan tampak seperti gambar 7 dibawah ini.



Gambar 7. Hasil *Running Script* yang ter-*encode*

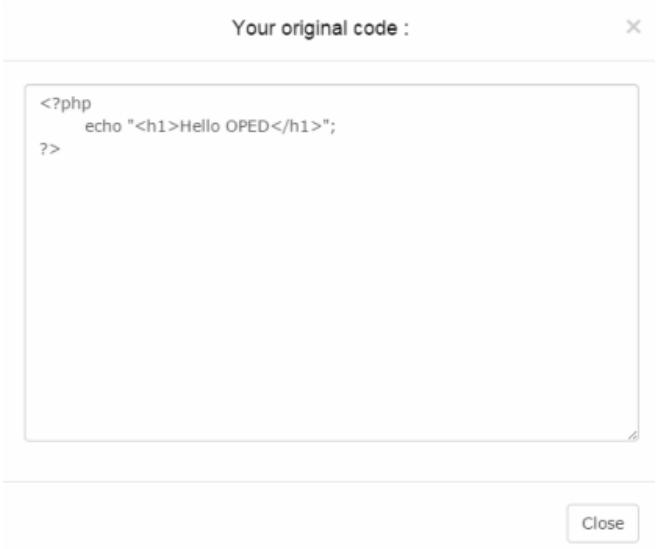
Pada gambar di atas terlihat kata “Hello OPED”, hal ini sesuai dengan kode PHP yang dilakukan proses *encode* yaitu “<?php echo "<h1>Hello OPED</h1>"; ?>”. Kemudian dilakukan pengujian proses *decoding* pada kode PHP yang telah ter-encode Gambar 7.



Gambar 8. Percobaan *Decode Script*

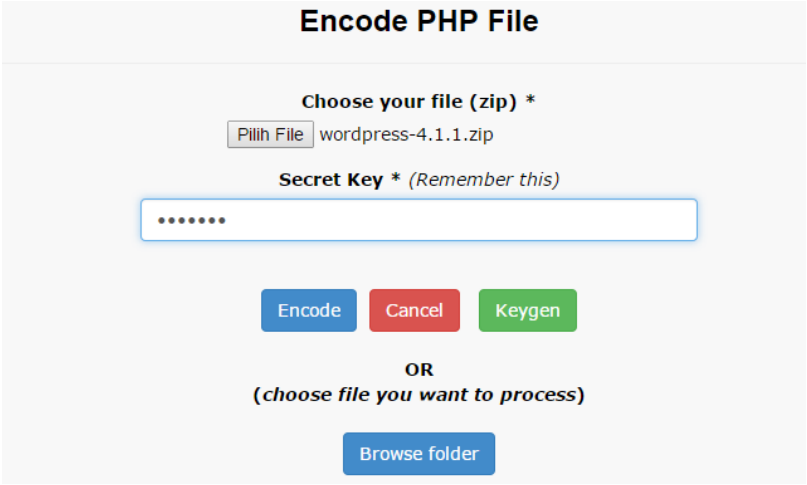
ISBN: 978-979-796-238-6

Dan berikut adalah hasil dari proses *decoding* kode PHP yang diinputkan pada gambar 8.



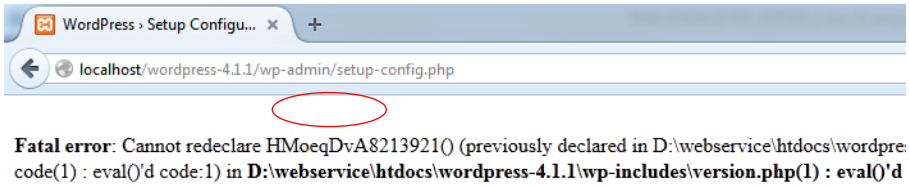
Gambar 9. Hasil *Decode Script*

Setelah memilih *file project* berupa file *archive* zip Gambar 9 kemudian tekan tombol *Encode* untuk langsung melakukan proses *encoding* keseluruh file PHP pada *file project* atau *Browse folder* untuk memilih file tertentu yang ingin dilakukan proses *encoding*. Pada pengujian ini proses *encoding* dilakukan keseluruh file PHP.



Gambar 10. Percobaan *Encode* Wordpress 4.1.1

Setelah memilih file wordpress Gambar 10, kemudian tekan tombol Encode untuk melakukan encoding keseluruh file PHP.



Gambar 11. *Running Code* Wordpress 4.1.1

Dari hasil running wordpress di atas terdapat kesalahan pada file version.php yang mengakibatkan wordpress tidak berjalan sebagaimana mestinya pada saat installasi.

ISBN: 978-979-796-238-6

Tabel 1. Hasil Pengujian *Encode*

No	Percobaan	Jenis pemrograman	Berhasil
1.	*Source code* PHP	PHP *Procedural*	Ya
2.	*File project* Taku	PHP *Procedural*	Ya
3.	Wordpress versi 4.1.1	PHP OOP *(Object Oriented Programming)*	Tidak

Kesimpulan

Berdasarkan analisa dari beberapa hasil pengujian yang telah dilakukan, maka kesimpulan yang didapatkan meliputi dengan adanya PHP *encoder* dan *decoder* ini seorang pengembang aplikasi PHP dapat menyembunyikan kode PHP yang sifatnya *private*. Integritas dari aplikasi yang telah dienkripsi akan lebih terjaga, karena kode PHP yang terenkripsi sulit diubah, Kelemahan-kelemahan sistem yang ada pada kode PHP secara otomatis tersembunyi, karena kode tidak dapat dibaca tanpa dilakukan proses dekripsi terlebih dahulu, Tidak memerlukan konfigurasi tambahan untuk menjalankan kode PHP yang sudah terenkripsi. Karena semua fungsi yang digunakan adalah fungsi bawaan PHP yang secara *default* aktif pada *webservice*. Sehingga sangat *support* untuk semua *server hosting*,Dapat melakukan proses *encode* dan *decode* pada *web file project*, sehingga memudahkan pengguna ketika ingin melakukan *encode* kesejumlah file PHP, Kunci keamanan dapat diset sesuai dengan keinginan pengguna.

3.1 Saran

Adapun beberapa saran yang mungkin bisa menjadi pandangan untuk pengembangan aplikasi PHP *encoder* dan *decoder* ini lebih lanjut yaitu :

1. Penggunaan algoritma lain yang mungkin lebih *secure* dan sulit untuk ditebak.
2. Penambahan fitur untuk melakukan proses *encoding* pada file PHP yang bercampur dengan HTML, CSS, JAVASCRIPT. Sehingga hanya kode PHP saja yang terenkripsi dengan ukuran file hasil *encoding* yang tidak terlalu membengkak atau mungkin sama dengan file aslinya.
3. Peletakkan kunci keamanan yang lebih sulit untuk ditebak. Mungkin dibuat sebuah file tertentu yang khusus untuk menyimpan kunci tersebut.

Daftar Pustaka

[1] Sholeh T. A, dkk. 2013. Mengamankan skrip pada bahasa pemograman PHP dengan menggunakan kriptografi Base64, Indonesia.
[2] Aprianto L. A. dan Winarno I. Rancang bangun PHP 5 Encoder, Indonesia.
[3] Wahyu C. F, dkk. 2012. *Penerapan Algoritma Gabungan RC4 Dan Base64 Pada Sistem Keamanan E-Commerce*, Jurnal Seminar Nasional Aplikasi Teknologi Informasi, ISSN 1907 - 5022.
[4] Setiawan O, dkk. 2014. *ALGORITMA ENKRIPSI RC4 SEBAGAI METODE OBFUSCATION SOURCE CODE PHP*, Indonesia.
[5] Chunlong Y, dkk. 2013. Security Analysis of PHP Encoder, China.
[6] Munir, R. 2006. Kriptografi. Bandung: Informatika.
[7] Welling L. and Thomson L. 2005. PHP and MySQL Web Development Third Edition, United States of America
[8] Doyle Matt. 2010. Beginning PHP 5.3, Indiana: Wiley Publishing.