Jurnal Computech & Bisnis, Vol. 2, No. 2, Desember 2008, 60-65
ISSN 1978-9629

KONSEP SOLUSI KEAMANAN WEB PADA PEMOGRAMAN PHP

KM. Syarif Haryana
STMIK Mardira Indonesia, Bandung 40235
kmsyarifharyana@yahoo.co.id

Abstract

PHP is an open source application and has the global auto facility on variables where each programmer is given the ease to apply. But this also facilitates ease attacker to destroy each program run . This attack can be avoided (minimized) by conducting safety. This paper will examine various aspects of the potential sources of easily disturbed security on PHP scripts and create alternative solutions workarounds.

In addition it also presented several vulnerabilities in the web in general and especially on the web are built using PHP scripts. Including some of the techniques that are usually used by intruders to download deface a web page.

At the end of the section presented several alternative solutions of various levels of information security including application level. The concept of web security solutions are a few web pages business owners , managers or administrators of web pages for secure web server and perform information security measures . Although the information security must be comprehensively and continuously due process deface web pages will be done when the inadvertence of the person in charge of the web page.

Keywords: *autoglobal, attacker, vulnerability, deface*

Abstrak

PHP adalah aplikasi yang open source dan mempunyai fasilitas *auto global* pada varibel dimana setiap programer diberikan kemudahan untuk mengaplikasikannya. Tetapi kemudahan ini pula yang memudahkan attacker untuk merusak setiap program dijalankan. Serangan ini dapat dihindari (diminimalkan) dengan cara mengadakan pengamannya. Untuklah tulisan ini akan mengkaji berbagai sumber tentang aspek potensial yang mudah diganggu keamanannya pada script PHP dan membuat alternatif solusi penangannya.

Selain itu dipaparkan pula beberapa vulnerability pada web umumnya dan khususnya pada web yang dibangun dengan menggunakan skrip PHP. Termasuk beberapa teknik-teknik yang biasanya digunakan oleh para penyusup untuk men-*deface* sebuah halaman web.

Pada bagian akhir dipaparkan beberapa alternatif pemecahan dari berbagai level keamanan informasi termasuk dari level aplikasi. Konsep solusi keamanan web

merupakan beberapa usaha pemilik halaman web, pengelola halaman web atau para admin web server untuk mengamankan dan melakukan langkah-langkah pengamanan informasi. Meskipun pengamanan informasi tersebut harus dilakukan secara komperhensif serta berkesinambungan karena proses *deface* halaman web akan dilakukan ketika terjadi kelengahan para penanggung jawab halaman web.

Kata Kunci: *Autoglobal, Attacker, Vulnerability, Deface*

A. PENDAHULUAN

Keamanan Web menjadi lebih lagi dibutuhkan dengan adanya kasus-kasus pencurian melalui Web, penipuan, perusakan, virus, worm, dan lain-lain. Karena pentingnya masalah kemanan ini maka bagi setiap orang yang ingin mengembangkan Webnya sudah selayaknya mempersiapkan diri sebaik-baiknya. Apalagi jika dalam pengembangan Web akan digunakan untuk aplikasi-aplikasi yang rentan atau kritis, maka kemanan yang baik akan menghindarkan kerugian yang mungkin didapat yang jumlahnya mungkin bisa sangat besar, baik secara material maupun nonmaterial.

B. ANALISIS MASALAH

Sebagai konsekuensi kepraktisan dan kemudahannya, instalasi default PHP banyak memiliki kelemahan keamanan. Variabel global di PHP dapat berasal dari masukan pengunjung Web (dari GET/POST/Cookie), sehingga bila programernya ceroboh tidak melakukan inisialisasi tiap variabel sebelum pemakaian, seorang penyerang dapat memasukkan nilai-nilai awal variabel ke dalam skrip untuk mengubah kelakuannya. Sebelum PHP 3.0.18 terdapat bug pada file upload yang banyak dieksploitasi untuk menembus banyak situs PHP. Dalam bug ini interpreter PHP dapat diakali untuk menulisi file di filesystem server mana pun sesuai keinginan penyerangnya, karena path dapat dimasukkan lewat form HTML.

Beberapa kelemahan ini dapat dikonfigurasi atau dimatikan. Karena itu seorang programer PHP dan admin perlu mengetahui opsi-opsi konfigurasi PHP agar sistem mereka lebih aman.

C. LUBANG KEAMANAN PHP

PHP dapat dijalankan sama seperti aplikasi CGI (*Common Gateway Interface*) seperti web server yang terintegrasi. Interpreter PHP mempunyai kemampuan untuk mengakses hampir semua *host-file system, network interfaces, IPC,* dan lain-lain. Konsekwensinya PHP potensial mendapat serangan dari attacker. Untuk meminimalkan serangan programmer harus menyadari dan mengetahui hal-hal yang tidak diharapkan (merusak) saat program dijalankan, yaitu pengetahuan kelemahan suatu sistem dan modus serangan secara umum yang diarahkan untuk mengganggu keamanan program tersebut. Lubang keamanan yang paling umum di dalam skrip PHP dan tak terkecuali pada aplikasi web yang manapun, adalah berkaitan dengan *User Input*. Banyak skrip menggunakan informasi *user* yang legal dalam bentuk format web dan memproses informasi ini dengan berbagai cara. Jika *user input* ini dilegalkan tanpa batasan, maka *user input* potensial menyisipkan perintah-perintah yang tidak diinginkan dalam skrip.

SQL Injections
Selain itu metode SQL Injections banyak pula digunakan untuk memanfaatkan kelemahan pada mesin server SQLnya, misalnya server yg menjalankan aplikasi tersebut. Hal ini dilakukan dengan mencoba memasukkan suatu script untuk menampilkan halaman error di browser, dan biasanya halaman error akan menampilkan *paling tidak* struktur dari hirarki server dan logika program. Metode ini memasukan "karakter" query tertentu pada sebuah "text area" Trend keamanan dan Serangan komputer| ver 1 atau di address browser dengan perintah-perintah dasar SQL seperti SELECT, WHERE, CREATE, UPDATE, dan lain-lain.

Cross site script (XSS)
Type lubang keamanan sistem lainnya yang biasa ditemukan di *web based applications* dengan melakukan *code injections* dengan malicious web pengguna kepada halaman web yang dilihat oleh user lainnya dimana memungkinkan penyerang untuk mencuri cookies, menipu user dengan memberikan credentials mereka, memodifikasi penampilan page, mengeksekusi seluruh sort dari malicious javascript code

Web Spoofing
• Web Spoofing
– Membuat web lain yang "*copy paste/ identik*" dari web asli – Membeli domain yang hampir identik
• Ex : klikbca.com (existing)
• Lalu dibeli domain clikbca.com / clickbca.com / clickbca.net
– Menangkap user dan password

Penanganan
• Digital Certificate
• Verisign, iTrust, …
• CA (Certificate Authority)
• https

D. PEMBAHASAN KONSEP/SOLUSI
Dari beberapa vulnerabilities pemograman aplikasi berbasis PHP yang telah di paparkan di atas, maka dapat diusulkan beberapa konsep solusi, yaitu :

Validasi Login
Tulisan ini diperuntukan bagi para newbie yang ingin membuat system login pada webnya dengan php. Apa yg akan dipaparkan berikut ini sebenarnya sangat umum dan dapat diperoleh dari berbagai sumber yang berhubungan dengan php. Berikut ini hal-hal yang harus kita pertimbangkan ketika membuat login:
1. Pastikan form login adalah form dari server kita.
2. Amankan input text untuk user dan password, metoda dan format data.
3. Hindari penggunaan register global (untuk php v 4.2.0 keatas sudah disable).
4. Expired time dari login yang dilakukan.
5. Pastikan file yang tidak boleh diakses tidak dapat dipanggil secara langsung.

Di bawah ini salah contoh satu logika dari banyak kemungkinan. Logika ini akan sangat beragam jadi bukanlah satu-satunya cara ataupun cara yang paling baik.
• Saat membuat form login (misalnya: index.php) sebagai default kita mulai dengan membuat session baru.
• Session_name disini sebagai referensi untuk session id di cookies dan URL.

Daftarkan suatu variabel session baru SES_TOKEN dengan memakai fungsi session_register.
• Buat token berupa kata acak yang akan kita gunakan untuk memastikan form adalah dari kita.

• Lalu Enkrip token agar lebih rumit. (Pada dasarnya token ini mirip dengan session id)
• Tambahkan variabel enkrip token tersebut dalam form melalui hidden.

Misalnya pada file "cekmasuk.php" kita lakukan pemeriksaan apakah data yang kita terima dari form yang kita buat sebelumnya. Amankan input text untuk user dan password, metoda dan format data. Pada saat kita menerima data maka sebelum kita olah misalnya untuk kasus user dan password, maka harus kita pastikan data tersebut tidak disisipi niat jahat. Untuk itu maka kita buat suatu fungsi filter. Hindari penggunaan register global (untuk php v 4.2.0 ke atas sudah disable/off).

Untuk hal ini kita dapat memperoleh data yang dikirim melalui predefine variabel milik php, yaitu:
$HTTP_GET_VARS
untuk metoda get
$HTTP_POST_VARS
untuk metoda post.

Expired Time dari login yang dilakukan. Setiap login yg dilakukan user sering kali mereka tidak melakukan logout, hanya mendiamkan atau malah meninggalkan ketika masih login. Oleh karena itu expired time ini adalah wajib dalam sistem login.

SQL Injection
Hilangkan Karakter-Karakter *Escape* dalam Perintah SQL
Kesalahan umum yang terjadi adalah penggunaan nilai variabel yang disediakan oleh *user* atau URL dalam sebuah *query* SQL tanpa menghilangkan karakter-karakter khusus. Perhatikan contoh kode fragmen berikut dari sebuah skrip yang dirancang untuk mengecek kebenaran *username* dan *password* yang dimasukkan dalam halaman HTML:

```
$query = “SELECT * FROM users
WHERE username=’ “ . $username
. “ ‘ AND password=’ “ .
$password . “ ‘ “;
// record yang memenuhi
perintah di atas terdapat di
suatu tempat
if (record_exists($query)) {
echo “Access granted”;
} else {
echo “Access denied”;
}
```

Perintah ini akan jalan jika pengaksesan menggunakan check.php?user name= admin&password= x. Akan tetapi, jika kode ini diakses dengan menggunakan check.php?username=admin& password=a%27+OR+1%3Di%271 (dan jika magic_quotes_gpc dibuat *disabled*) maka *password* akan menjadi Password=’a’ or 1=’1’ sehingga *record* pengguna admin akan selalu dikembalikan
berapapun nilai password.

Penggunaan HTTPS Protokol
Solusi selanjutnya kita dapat pergunakan protokol yang dapat mendukung segi keamanan yaitu https. **HTTPS** (HTTP melalui SSL or HTTP Secure), merupakan protokol HTTP yang menggunakan Secure Socket Layer **(SSL)** atau Transport Layer Security (TLS) sebagai sub layer dibawah HTTP aplikasi layer yang biasa. HTTP di enkripisi dan deskripsi dari halaman yang diminta pengguna serta halaman yang dikembalikan oleh web server. HTTPS digunakan untuk melindungi dari orang mengakses tanpa izin dan dari serangan *man-in-themiddle.*HTTPS dikembangkan oleh Netscape.

Dengan HTTPS kita dapat melakukan proteksi data yaitu hanya penerima saja yang dapat membaca data, Kenyamanan (data privacy), memungkinkan identifikasi server ataupun client, otentikasi server dan klien, dan integritas data. Sedangkan **SSL** (Secure Socket Layer) adalah arguably internet

yang paling banyak digunakan untuk enkripsi. Ditambah lagi, SSL sigunakan tidak hanya keamanan koneksi web, tetapi untuk berbagai aplikasi yang memerlukan enkripsi jaringan end-to-end.

HTTPS, TLS, and SSL

Https adalah versi aman dari HTTP, protokol komunikasi dari World Wide Web menyediakan autentikasi dan komunikasi tersandi dan penggunaan dalam komersi elektris. Pendekatan HTTPS sangatlah simpel, Client membuat koneksi ke server, melakukan negosiasi koneksi SSL, kemudian mengirim HTTP tersebut melalui aplikasi SSL. Deskripsi ini menjadikannya terlihat mudah. Selain menggunakan komunikasi plain text, HTTPS menyandikan data sesi menggunakan protokol SSL (Secure Socket layer) atau protokol TLS (Transport Layer Security). Kedua protokol tersebut memberikan perlindungan yang memadai dari serangan eavesdroppers, dan man in the middle attacks.

Pada umumnya port HTTPS adalah 443. Terdapat perbedaan *port* yang spesifik, HTTPS menggunakan *port* 443 sedangkan HTTP menggunakan *port* 80 dalam berinteraksi dengan *layer* yang dibawahnya, TCP/IP/ HTTPS dan SSL mendukung penggunaan dari X.509 sertifikat digital dari server, sehingga jika diperlukan, pengguna dapat mengotentikasi pengirimnya. Kecuali perbedaan *port* yang spesifik, HTTPS menggunakan *port* 443 sedangkan HTTP menggunakan *port* 80 dalam berinteraksi dengan *layer* yang dibawahnya, TCP/IP/ Tingkat keamanan tergantung pada ketepatan dalam mengimplementasikan pada browser web dan perangkat lunak server dan didukung oleh algorithma penyandian yang aktual. Oleh karena itu, pada halaman web digunakan HTTPS, dan URL yang digunakan dimulai dengan ‘https://’ bukan dengan ‘http://’ Efektifitas dari HTTPS dapat dibatasi oleh kurangnya implementasi dari browser atau perangkat lunak server atau kurangnya dukungan dari beberapa algoritma.

Selanjutnya, walapun HTTPS dapat mengamankan perjalanan data antara server dan klien, setelah data didekripsi tujuannya, itu hanya aman sebagai *host* komputer. Kesalahpahaman yang sering terjadi pada pengguna kartu kredit di web ialah dengan menganggap HTTPS “sepenuhnya” melindungi transaksi mereka. Sedangkan pada kenyataannya, HTTPS hanya melakukan enkripsi informasi dari kartu mereka antara browser mereka dengan web server yang menerima informasi. Pada web server, informasi kartu mereke secara tipikal tersimpan di database server (terkadang tidak langsung dikirimkan ke pemroses kartu kredit), dan server database inilah yang paling sering menjadi sasaran penyerangan oleh pihak-pihak yang tidak berkepentingan

E. KESIMPULAN

Kesimpulan yang dapat diambil dari pembahasan makalah ini antara lain sebagai berikut:

1. Teknologi Web server merupa-kan inti setiap desain aplikasi Web. Tanpa memberi perhatian khusus pada keamanannya, konfigurasi defaultnya justru akan menjadi sejumlah jalan penyerangan bagi para penye-rang. Konfigurasi default biasa-nya memberikan berbagai ke-mudahan dan fitur tambahan, namun tidak selalu diperlukan. Untuk itu perlu diperhatikan sekali lubang keamanannya.

2. Dalam mengembangkan aplikasi Web, sangat penting untuk dipa-hami bagaimana kerja masing-masing teknologi Web yang di-pakai

agar dapat diantisipasi kelemahan-kelemahan keama-nannya. Seperti :

- Programming Language Vulnerabilities (PHP, .NET)
- SQL Injections of hostile SQL commands allow attackers to steal data
- Cross-Site Scripting Exploits allow attackers to insert hostile content from domains that they control.

3. Mengapa HTTPS :
   - Melindungi data dari akses yang tidak diijinkan, hanya penerima yang diijinkan untuk membaca data
   - Menjaga kerahasiaan data (data privasi).
   - Integritas data
   - Klien dan server autentikasi
   - Memastikan bahwa tidak ada yang bisa merusak data yang ditransmisikan.

DAFTAR PUSTAKA

Clancy Malcolm, *Ten Security Check For PHP*, Website, http://www.onlamp.com/pub/a/php/2003/03/20/php_security.htm

Jordan Dimov,*On The Security Of PHP*, Website : http://www.developer.com/lang/php/ article.php/922871

Sufehmi, Harry, *Security di PHP*, Website http://www.tf.itb.ac.id/~eryan/Php/ PHPSecurity.txt

John Coggeshall ,*PHP Security*,Website http://www.onlamp/pub/au/135

Kadir, Abdul, *Dasr Penmrograman Web Dinamis Menggunakan PHP*, Andi, Yogyakarta, 2002.

An ISS Technical White Paper, Web Application Protection