ANALISIS *WEB VULNERABILITY* PADA *DIGITAL LIBRARY SERVER* UNIVERSITAS BINA DARMA

Ilman Zuhri Yadi[1], Muhammad Izman Herdiansyah[2]

Sistem Informasi, Fakultas Ilmu Komputer, Universitas Bina Darma
Jalan Jenderal Ahmad Yani No.12 Palembang
[1] ilmanzuhriyadi@mail.binadarma.ac.id, [2] herdians11816@gmail.com

Abstrak

Dynamic web page technology has become part of the development of the Internet. Dynamic website content that can show different results in each user in accordance with the desired configuration and needs. But behind the advantage that this technology has vulnerabilities. One threat is the use of scripts to gain access to the system by using cookies. Techinque Cross-Site Scripting (XSS) and SQL Injection is a form of attack from the above techniques. Application Server Bina Darma University Digital Library has experienced the problem.
The problem in this research, there are still weaknesses in web applications and there has been no improvement scripting program. Tools used in this study is Acunetix Vulnerability Scanner version 6.5 and Shadow Database Scanner version 7.75 for security analysis of web and database servers. The research method used is Action Research. This research was successfully conducted an analysis of security system includes a web server, applications programs and database servers on the system Digital Library of Bina Darma University.

Kata kunci : *Cross-Site Scripting, SQL Injection, Web Vulnerability, Digital Library, Acunetix Vulnerability Scanner, Shadow Database Scanner*.

1. Pendahuluan

Teknologi dynamic web page kini telah menjadi bagian yang tidak terpisahkan dari perkembangan teknologi informasi khususnya di dunia maya atau dikenal dengan Internet. Isi website yang dinamis akan dapat menampilkan hasil yang berbeda disetiap pengguna sesuai dengan konfigurasi dan kebutuhan yang diinginkan. Teknologi ini membawa perubahan yang signifikan dalam proses pembangunan sistem penyedia layanan dalam jaringan internet. Teknologi ini memampukan penyedia layanan untuk memberikan layanan yang lebih inovatif. Efek yang diharapkan tentu saja peningkatan dari segi ekonomi.

Namun dibalik keuntungan itu semua, teknologi ini memiliki permasalahan dari segi keamanan. Salah satu ancaman yang paling umum selain *virus*, *trojan, backdoor* dan *spam* adalah pemanfaatan *scripts* untuk memperoleh akses kedalam sistem. Penggunaan *cookie* dalam *dynamic web page* telah banyak dijumpai, di antaranya untuk menyimpan asosiasi unik dari *account* pengguna. Beberapa situs *web* seperti Yahoo, Hotmail maupun Netscape dapat dijadikan contoh pemanfaatan tersebut. Selain situs tersebut, beberapa situs perdagangan elektronik juga menggunakan *cookie* untuk menempatkan identitas unik pengguna bagi keperluan autentifikasi maupun autorisasi pada situs yang menggunakan skenario *log on*. Biasanya digunakan dua token autentifikasi, yaitu nama pengguna dan kata sandi (*password*), kedua token ini kemudian disimpan dalam *cookie* untuk memudahkan identifikasi atas banyaknya pengguna, juga untuk keperluan pelakuan batas sesi koneksi ke situs tersebut. Teknik *Cross Site Scripting* (XSS), merupakan teknik yang banyak digunakan bagi keperluan mendapatkan *cookie* ini. Begitu *cookie* didapatkan, si penyerang ini akan dapat memuat nilai *cookie* curian tersebut, lalu mengarahkan *browser*-nya ke situs aplikasi yang menggunakan *cookie* tersebut. Dan mengakses *account* korban, tanpa perlu menghabiskan waktu untuk memecahkan sandi dan enkripsi atas kombinasi pengguna dan kata sandi.

Selain itu bentuk penyerangan lainnya yang memanfaatkan *scripting* adalah *SQL Injection*. *SQL Injection* adalah salah satu tipe meng-*hack* yang hanya membutuhkan *port* 80 dan tidak memerlukan *port* lain. *SQL Injection* akan menyerang aplikasi *web* yang berbasiskan *side-server scripting* seperti ASP, JSP, PHP, CGI, dan yang mirip dengan itu. *SQL Injection* merupakan teknik untuk mengeksporasi aplikasi *web* dengan memanfaatkan suplai data dari *client* dalam *SQL syntax*. Banyak

halaman *web* memakai parameter dari *web user* untuk menggunakan *query* ke dalam *database*. Sebagai contoh ketika *user* akan *login*, halaman *user* akan mengirim *user* dan *login* sebagai parameter untuk digunakan sebagai *SQL* dan mencek apakah *user* dan *password* cocok. Dengan *SQL Injection* ini sangat mungkin untuk kita mengirim *username* dan *password* dan dianggap benar.

Bentuk penyerangan dari kedua metode diatas yaitu *Cross Site Scripting* dan *SQL Injection* pernah dialami oleh *server Digital Library* Universitas Bina Darma. Selama 1 hari *hacker* berhasil men-*deface* situs *Digital Library* (http://digilib.binadarma.ac.id), dan merubah wajah dari halaman *web*-nya. Sebagai solusinya pada saat itu adalah dengan mengatur ulang seluruh konfigurasi dan tatanan *database* pendukung di server Digital Library, serta menon-aktifkan *Internet Protocol* (IP) *Address Public*. Sebuah konsekuensi yang pelik diambil, demi menyelamatkan reputasi dan nama baik. Hal ini tidak bisa dibiarkan berlarut-larut, harus ada perbaikan pada aplikasi *digital library*-nya dan keamanan *website* bisa ditingkatkan. Hal lain yang menjadi bahan pertimbangan untuk diperbaiki *website* ini adalah untuk mendukung Webometric, dikarenakan salah satu penilaiannya adalah *Rich Files* (R) / Volume *file* yang ada di situs Universitas. Dimana format file yang dinilai layak masuk di penilaian (berdasarkan uji relevansi dengan aktivitas akademis dan publikasi) adalah: *Adobe Acrobat* (.pdf), *Adobe PostScript* (.ps), *Microsoft Word* (.doc) dan *Microsoft Powerpoint* (.ppt). Data-data ini diambil menggunakan Google dan digabungkan hasil-hasilnya untuk setiap jenis berkas.

Untuk itulah pada penelitian ini nantinya akan dilakukan analisis *web vulnerability* khususnya pada *content website Digital Library* Universitas Bina Darma. Dan melakukan beberapa perbaikan *script* di *server Digital Library*, untuk mengatasi celah-celah keamanan di sistem tersebut.

2. Metodologi Penelitian

Dalam penelitian ini metode penelitian yang di gunakan adalah penelitian tindakan atau a*ction research,* dalam penelitian tindakan mendeskripsikan, menginterpretasi dan menjelaskan suatu situasi sosial pada waktu yang bersamaan dengan melakukan perubahan atau intervensi dengan tujuan perbaikan atau partisipasi.

Davison, Martinsons dan Kock (2004, dalam Chandrax 2008), menyebutkan penelitian tindakan, sebagai metode penelitian, didirikan atas asumsi bahwa teori dan praktik dapat secara tertutup diintegrasikan dengan pembelajaran dari hasil intervensi yang direncanakan setelah diagnosis yang rinci terhadap konteks masalahnya. Lima tahapan yang merupakan siklus dari a*ction research*,

1. Melakukan diagnosa (*Diagnosing*)
2. Membuat rencana tindakan (*Action Planning*)
3. Melakukan tindakan (Action Taking)
4. Melakukan evaluasi (Evaluating)
5. Pembelajaran (L*earning*)

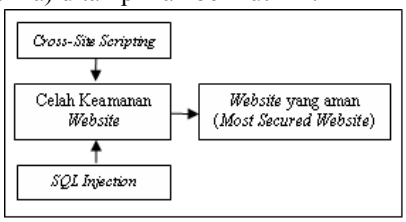
a. Kerangka Pemikiran

Dalam kerangka pemikiran penelitian ini parameter yang akan diukur terdiri dari :

a. ***Cross-Site Scripting (XSS)***, yang dijadikan sebagai parameter pada penelitian ini adalah dapat terjadi ketika halaman *web* yang dinamis menampilkan hasil yang tidak *valid*. Hal ini memudahkan penyerang untuk menulis perintah-perintah tertentu untuk menghasilkan *output* baru yang dapat dieksekusi pada halaman *web* lain.
b. ***SQL Injection***, yang dijadikan sebagai parameter pada penelitian ini adalah sebuah teknik untuk mengeksporasi aplikasi *web* dengan memanfaatkan suplai data dari *client* dalam sintak SQL. Kedua parameter diatas yang sebagian besar mendominasi kelemahan dari situs *Digital Library* Universitas Bina Darma setelah diadakan pengujian menggunakan *tools Acunetix Vulnerability Scanner*.
c. **Celah Keamanan *Website***. Pada bagian ini diadakan serangkaian pengujian akses ke *website Digital Library* (versi GDL 4.0) dan melakukan proses pengecekan kelemahan (*vulnerability*) *website* dan database server-nya menggunakan *tools Acunetix Vulnerability Scanner* dan *Shadow Database Scanner.* Dari hasil pengujian itu menghasilkan laporan *Acunetix Web Audit* sebagai bahan acuan untuk melakukan perbaikan terhadap *web server*, program aplikasi dan *database server* sistem ini dari beberapa celah keamanan yang ditemukan pada *Digital Library Server*.
d. ***Website* yang aman (*Most Secured Website*)**. Pada bagian ini dilakukan analisa hasil audit dari *tools Acunetix Vulnerability Scanner* dan *Shadow Database Scanner.* Dengan melakukan proses perbaikan dengan proses *upgrade* (dari versi GDLN 4.0 ke GDLN 4.2) dan *update* terhadap *web server*, program aplikasi dan *database server Digital Library* berdasarkan laporan celah keamanan dari *software Acunetix Web Audit*. Serta melakukan proses migrasi data dari GDLN 4.0 ke GDLN 4.2. Melakukan proses analisis celah keamanan lagi terhadap *web server*, program aplikasi dan *database server* pada *Digital Library* yang telah di *upgrade* ke versi GDLN 4.2 dan *update* ke versi terbaru dari Apache, PHP dan MySQL. Untuk mendapatkan hasil yang maksimal yaitu tidak didapatkannya lagi celah keamanan atau masuk ke kategori *level* 1 (*Low*). Dimana pada level ini artinya celah-celah yang memungkinkan terjadinya ancaman dan akses *ilegal* yang berpotensi merusak sistem sudah di minimalisir.

Kerangka pemikiran untuk Analisis *Web Vulnerability* untuk meningkatkan Keamanan

Website (studi kasus : *Digital Library* Universitas Bina Darma) ditampilkan berikut ini :



Gambar 1. Kerangka Pemikiran

2.1. *Web Security*

Di masa lalu, tujuan utama dari komputer adalah untuk menyimpan informasi yang diperlukan oleh organisasi untuk kegiatan harian dari organisasi itu sendiri. Komputer hanya digunakan sebagai alat pusat pengolahan data saja, dan terbatas hanya untuk penggunaan di internal organisasi saja. Oleh karena itu, ancaman keamanan komputer biasa dan pada dasarnya berhubungan dengan staf di organisasi (misalnya: penyalahgunaan *account*, pencurian, atau data manipulasi oleh para pengguna). Menangani ancaman ini mudah sekali bagi organisasi, dikarenakan hampir tidak ada kemungkinan ancaman dari eksternal. Ancaman ini umumnya ditangani dengan dengan menjaga komputer dengan informasi penting di sebuah ruangan khusus dan terisolasi, serta secara manual di verifikasi bahwa data pada komputer belum dirusak.

Namun, penggunaan komputer sejak dari awal hingga saat ini telah berubah secara radikal. Sekarang organisasi dalam menggunakan komputer untuk menyimpan data yang dapat diakses dari lokasi mana saja di dunia ini. Selain itu, komputer tidak lagi digunakan hanya dalam organisasi. Komputer banyak digunakan oleh individu dan pemakai rumah tangga untuk berkomunikasi lebih cepat di seluruh dunia. Karena penggunaan yang luas seperti itu, ancaman keamanan komputer secara alami telah meningkat. Banyak ancaman keamanan terjadi dalam bentuk pencurian *virtual* di *Internet*. Dalam hitungan detik, seorang pencuri *virtual* dapat mengakses sistem dan mencuri informasi penting, seperti *password* dan nomor kartu kredit. Kerusakan juga bisa dilakukan dengan infiltrasi sistem dan informasi tentang itu dengan melewatkan *virus* dan *worm*.

Saat ini, *Internet* telah menjadi media di mana orang dapat terhubung. Ini adalah *platform* di mana jutaan komputer, seluruh saham, dunia dan akses informasi. Transaksi bisnis *e-commerce* seperti pasar *online*, kini menjadi kenyataan. Namun dengan evolusi dari *Web* dan peningkatan penggunaan dalam setiap aspek kehidupan, kebutuhan akan keamanan *web* sudah menjadi keharusan.

Ada kekhawatiran beberapa kunci yang terkait dengan keamanan *Web*: Seberapa aman sistem yang mengontrol pertukaran informasi di *Web*? Seberapa aman informasi yang disimpan pada banyak komputer di seluruh *Web*? Ini adalah fakta diketahui bahwa apa yang dapat digunakan juga dapat disalahgunakan. Sebagai contoh, *e-commerce* telah membuat hidup kita lebih mudah, namun ada beberapa risiko yang melekat. Mengikuti alur pemikiran ini, kita perlu rencana keamanan *Web* dalam suatu organisasi baik di tingkat sistem dan data. Keamanan di tingkat sistem memastikan bahwa sistem Anda tidak *hacked* sejauh itu jatuh. Keamanan di tingkat data menjamin bahwa informasi pada sistem Anda tidak dirusak.

Harus selalu ingat bahwa jika informasi organisasi adalah *hacked* baik melalui jaringan atau melalui cara lain, bisa dikenakan biaya berat untuk perusahaan. Sebuah kegagalan dalam keamanan jaringan juga bisa biaya organisasi dalam hal *goodwill* dan reputasinya. Tidak ada organisasi lain akan tertarik dalam melakukan bisnis dengan organisasi yang tidak bisa melindungi informasi dan sistem keamanan.

Sebuah pelanggaran keamanan dapat didefinisikan sebagai akses ilegal terhadap informasi yang dapat mengakibatkan pengungkapan, penghapusan, atau perubahan informasi. Dengan kata lain, suatu pelanggaran keamanan terjadi ketika informasi atau sistem yang digunakan atau diakses untuk tujuan ilegal. Sebuah pelanggaran keamanan *web* dapat berlangsung dalam beberapa bentuk, seperti pelanggaran di jaringan organisasi, *hacking* ke dalam sistem atau jaringan, perubahan informasi organisasi atau individu, serangan virus, gangguan atau penolakan layanan, perusakan, dan pencurian. Berikut ini adalah beberapa jenis umum pelanggaran keamanan : (Shweta Bhasin, 2003)

a. ***Accessing subscriber details to send spam e-mail***
   Untuk mempromosikan penawaran produk baru, perusahaan kartu kredit biasanya mengakses informasi pelanggan dari *database* penyedia layanan *e-mail*. Hal ini tentu saja tanpa sepengetahuan penyedia layanan *e-mail*.
b. ***Unauthorized access of confidential data to create fraudulent identities***
   Seseorang mengakses rincian tambahan, seperti alamat tempat tinggal, nomor kontak, nomor jaminan sosial, dan detail nomor rekening dari *database* bank untuk menciptakan identitas palsu.
c. ***Eavesdropping***
   Sebuah badan intelijen suatu negara terhubung ke jaringan negara lain untuk mengakses informasi pertahanan yang sensitif dan rahasia.
d. ***Promoting your organization on somebody else's Web site***
   Sebuah organisasi membuat sebuah *server* untuk *host* situs di *Web*. Organisasi lain yang serupa mengakses *web server* ini dengan cara ilegal dan *host* beberapa halaman *web* di situs untuk mempromosikan organisasinya.
e. ***Using an automated script to try to log in to a computer system***

Seorang *hacker* menggunakan *script* otomatis untuk membuat berbagai upaya untuk *login* ke sistem komputer. Akibatnya, pengguna jasa yang berwenang *logon* ditolak oleh komputer karena komputer sedang sibuk karena menolak permintaan dari *hacker*.

f. ***Gaining unauthorized access to a mail server***
Akses yang tidak sah oleh seseorang untuk keuntungan *pribadi* ke *mail server* organisasi untuk mengirim dan menerima pesan *e-mail*.

g. ***Gaining unauthorized access to the network to gain information***
Seseorang menyusup ke jaringan bank atau perusahaan keuangan untuk mentransfer sejumlah besar uang ke rekening fiktif.

h. ***Virus attacks***
Virus biasanya menyebar melalui pesan *e-mail*. Sebuah serangan virus juga dapat terjadi dalam jaringan organisasi, dan melalui jaringan yang menyebar melalui *Internet.*

i. ***DNS hijacking***
Domain Name System (DNS) adalah *database* yang peta nama domain ke alamat IP. Komputer yang terhubung ke *internet* menggunakan DNS untuk menyelesaikan URL ke alamat IP dari situs yang perlu diakses. Dalam pembajakan DNS, para *hacker* mendapatkan akses ke layanan DNS dan membuat perubahan dalam informasi yang memetakan nama domain ke alamat IP. Karena ini, pengguna akan diarahkan ke situs yang berbeda dari yang mereka inginkan untuk mengakses.

j. ***DoS attacks***
Denial-of-service (DoS) serangan adalah serangan berbasis jaringan di mana pengguna resmi ditolak penggunaan layanan jaringan. Serangan DoS terjadi karena berbagai alasan, seperti penggunaan sumber daya yang tidak sah. Sebuah contoh umum serangan DoS adalah pengguna yang tidak sah menggunakan lokasi FTP Anda untuk meng-*upload* volume data yang besar. Hal ini menyebabkan penyumbatan yang tidak perlu di ruangan penyimpanan dan menghasilkan lalu lintas jaringan yang ramai.

k. ***DDoS attacks***
Distributed denial-of-service (DDoS) serangan adalah bentuk canggih dari serangan DoS. Dalam serangan DDoS, sistem target diserang dari beberapa komputer di *Internet*. Tanpa pengetahuan pemilik, *hacker* menciptakan suatu aplikasi dan tempat aplikasi di beberapa lokasi di *Internet*. Aplikasi ini tidak terdeteksi, karena mereka tidak membahayakan sistem di mana mereka berada. Ketika serangan diluncurkan, sistem target terganggu dari semua komputer yang berbeda yang memiliki aplikasi yang diinstal oleh *hacker*.

2.2. *Web Vulnerability*

World Wide Web (WWW atau *Web1*) merupakan salah satu "*killer applications*" yang menyebabkan populernya *Internet*. WWW dikembangkan oleh Tim Berners-Lee ketika bekerja di CERN (Swiss). Sejarah dari penemuan ini dapat dibaca pada buku karangan Tim Berners-Lee ini. Kehebatan *Web* adalah kemudahannya untuk mengakses informasi, yang dihubungkan satu dengan lainnya melalui konsep *hypertext*. Informasi dapat tersebar di mana-mana di dunia dan terhubung melalui *hyperlink*. Informasi lebih lengkap tentang WWW dapat diperoleh di *web* W3C <http://www.w3.org>. (Budi, 2002)

Berkembangnya WWW dan *Internet* menyebabkan pergerakan sistem informasi untuk menggunakannya sebagai basis. Banyak sistem yang tidak terhubung ke *Internet* tetapi tetap menggunakan basis *Web* sebagai basis untuk sistem informasinya yang dipasang di jaringan *Intranet*. Untuk itu, keamanan sistem informasi yang berbasis *Web* dan teknologi *Internet* bergantung kepada keamanan sistem *Web* tersebut.

Arsitektur sistem *Web* terdiri dari dua sisi: *server* dan *client*. Keduanya dihubungkan dengan jaringan komputer (*computer network*). Selain menyajikan data-data dalam bentuk statis, sistem Web dapat menyajikan data dalam bentuk dinamis dengan menjalankan program. Program ini dapat dijalankan di server (misal dengan CGI, servlet) dan di *client* (applet, Javascript). Sistem *server* dan *client* memiliki permasalahan yang berbeda. Keamanan *server* WWW biasanya merupakan masalah dari seorang administrator. Dengan memasang *server* WWW di sistem anda, maka anda membuka akses (meskipun secara terbatas) kepada orang luar. Apabila server anda terhubung ke *Internet* dan memang *server* WWW anda disiapkan untuk publik, maka anda harus lebih berhati-hati sebab anda membuka pintu akses ke seluruh dunia!

Adanya lubang keamanan di sistem WWW dapat dieksploitasi dalam bentuk yang beragam, antara lain : (Budi, 2002)

a. Informasi yang ditampilkan di *server* diubah sehingga dapat mempermalukan perusahaan atau organisasi anda (dikenal dengan istilah *deface*1);
b. Informasi yang semestinya dikonsumsi untuk kalangan terbatas (misalnya laporan keuangan, strategi perusahaan anda, atau *database client* anda) ternyata berhasil disadap oleh saingan anda (ini mungkin disebabkan salah *setup server*, salah *setup router / firewall*, atau salah *setup authentication*);
c. Informasi dapat disadap (seperti misalnya pengiriman nomor kartu kredit untuk membeli melalui WWW, atau orang yang memonitor kemana saja anda melakukan *web surfing*);
d. *Server* anda diserang (misalnya dengan memberikan *request* secara bertubi-tubi)

sehingga tidak bisa memberikan layanan ketika dibutuhkan (*denial of service attack*);

e. Untuk server *web* yang berada di belakang *firewall*, lubang keamanan di *server web* yang dieksploitasi dapat melemahkan atau bahkan menghilangkan fungsi dari *firewall* (dengan mekanisme *tunneling*).

Berikut ini menurut Lauri Auronen (2002) beberapa celah keamanan yang biasa menyerang sistem keamanan di WWW yang dapat dikelompok sebagai berikut :

a. ***Detecting backend systems*.** Penggunaan kode template, komentar dalam kode *HyperText Markup Language* (HTML) atau bahkan bentuk *Uniform Resource Locator* (URL) yang digunakan dalam aplikasi *web* menyediakan informasi tentang sistem *backend* atau lingkungan pengembangan aplikasi *web*. Terutama penggunaan kode *template* dapat membuktikan berbahaya (NALNEESH, 2000). Kode ini dapat tersedia secara luas dan dapat mengandung *bug* yang mudah dianalisa dari *source code*. Informasi ini dapat digunakan untuk mengeksploitasi kerentanan khusus atau mempersempit fokus pada pembacaan kerentanan yang dilakukan pada sistem.

b. ***Session hijacking*.** HTTP adalah *stateless*. Aplikasi *Web* adalah sering perlu untuk tindakan pengguna mengikat ke *stateful* sesi tunggal. HTTP dibuat *stateful* dengan membuat objek sesi pada sisi server dan menyimpan *identifier* obyek ini, sesi disebut *identifier*, dalam sebuah cookie di *browser* klien atau sebagai parameter berlalu dalam URL setiap permintaan klien. Mengubah sesi pengenal di *cookie* atau URL pada sisi *client* agar sesuai dengan identifier sesi pengguna lain dapat digunakan untuk membajak sesi ini (NALNEESH, 2000). Pengidentifikasi sesi ini dapat diketahui dengan mendengarkan lalu lintas jaringan atau dengan menebak.

c. ***Cookie poisoning.*** *Cookie* dalam beberapa aplikasi berisi informasi bisnis aktual tertentu, seperti item disimpan dalam *shopping cart* dan daftar harga mereka. Informasi ini dapat dengan mudah berubah dan jika tidak ada mekanisme otentikasi yang berada di tempat, untuk memeriksa validitas *cookie* di sisi *server*, ini mengarah kepada kompromi dari aplikasi. (NALNEESH, 2000)

d. ***Form manipulation*.** *Form* HTML dapat disimpan ke dalam disk pada sisi *client* dan dapat di *edit*. Hal ini menjadi masalah jika ada *field* tersembunyi dalam bentuk yang berisi data yang dianggap tidak berubah, seperti harga *item*. Sekali lagi, jika mekanisme otentikasi yang cukup yang tidak pada tempatnya, ini menyebabkan serangan mungkin terjadi (NALNEESH, 2000). Bidang Formulir juga dapat memiliki kendala seperti panjang maksimum. Perubahan ini dapat menyebabkan *buffer overflows* dalam aplikasi *web*.

e. ***URL parameter tampering***. Aplikasi *Web* sering mengambil parameter sebagai bagian dari URL yang dikirim oleh *browser*. Serangan terhadap parameter URL adalah serangan paling mudah perusakannya, karena setiap pengguna dapat mengklik pada *address bar browser* mereka dan di ketik ke dalam parameter baru.

f. ***HTTP header modification.*** *Header* HTTP digunakan untuk melewati beberapa variabel antara agen pengguna dan *server web*. Variabel ini termasuk *cookie* yang telah diatur oleh situs, URL pengarah dan bahasa dari agen pengguna. Seperti formulir isian data, variabel-variabel ini juga dapat secara bebas dimodifikasi oleh pengguna dengan alat yang cocok. Ini dapat digunakan untuk, misalnya, *cross-site scripting* dan *SQL injection query.*

g. ***Bypassing intermediate forms in a multiple-form set*.** Karena HTTP adalah *stateless*, hal itu bisa mungkin bisa juga tidak untuk menjamin bahwa halaman yang diakses dalam urutan yang telah ditentukan. Pengguna bisa menebak dan ketik alamat halaman lain. Hal ini menimbulkan masalah dalam bentuk *multiple-set* jika bentuk yang kemudian bergantung pada masukan yang diberikan dalam bentuk sebelumnya. Hal ini dapat menyebabkan masalah pada fungsionalitas dari aplikasi *web* itu sendiri atau kompromi dari *platform* yang mendasari, biasanya melalui *buffer overflow*. (NALNEESH, 2000)

h. ***SQL Injection Queries.*** Masukan yang diberikan oleh user dalam bentuk yang dikirim untuk diproses ke aplikasi *backend*. Jika bidang ini tidak cukup di validasi, mereka mungkin berisi karakter dengan arti khusus dalam SQL. Jika masukan tersebut kemudian digabungkan sebagai bagian dari query SQL karakter khusus dapat digunakan untuk membangun sebuah *query* SQL yang sudah di modifikasi, maka dapat mengarah ke modifikasi informasi yang bocor dan berbahaya dari *database*. (NALNEESH, 2000)

i. ***Cross-Site Scripting.*** Ketika pengguna dapat mengirim informasi ke aplikasi web melalui beberapa mekanisme dan informasi yang kemudian ditampilkan untuk pengguna lain, adalah kemungkinan untuk dimasukan *script* HTML yang berbahaya. Kode ini, ketika ditampilkan ke pengguna lain, akan tampak berasal dari aplikasi *web* itu sendiri. Kode kemudian dapat digunakan untuk berbagai serangan informasi kompromi yang rahasia. Daftar serangan menggunakan *cross-site scripting* (XSS) dapat ditemukan di CERT *advisory* . (Mike dkk, 2009).

j. ***3rd Party Software Misconfiguration*.** Perangkat lunak terkonfigurasi menyebabkan berbagai masalah. Biasanya *misconfigurations* ini

memungkinkan beberapa serangan aplikasi *web* lainnya. Gaur, dalam artikelnya, memberi contoh yang *misconfiguration* dapat menyebabkan daftar direktori yang tidak sah.

k. ***Forceful Browsing.*** Dapat berarti membuat beberapa permintaan ke *server web* dengan pola URL komponen aplikasi *web* khas seperti program CGI. Teknik ini dapat digunakan untuk mendapatkan informasi tentang aplikasi *web.*

Dalam penelitian ini nantinya, celah keamanannya akan dititikberatkan kedalam bentuk serangan *Cross-Site Scripting* (XSS) dan *SQL Injection* yang pernah menyerang *Digital Library Server* Universitas Bina Darma.

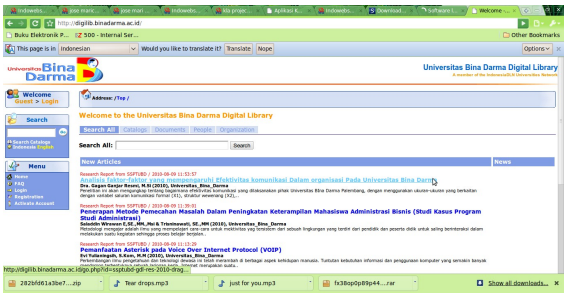
3. Hasil

Aplikasi *Digital Library* Universitas Bina Darma dikembangkan melalui proyek TPSDP tahun 2004. Aplikasi *Digital Library* ini dirancang agar dapat dimanfaatkan oleh dosen dan mahasiswa serta masyarakat umum yang menjadi anggota *Digital Library* bisa secara mudah untuk mengakses koleksi-koleksi *digital* yang tersedia di perpustakaan *Digital* Universitas Bina Darma. Koleksi-koleksi *digital* yang tersedia baik berupa hasil-hasil penelitian dalam bentuk Skripsi / Tugas Akhir, thesis ataupun jurnal, dan koleksi *ebook* yang ada bisa dimanfaatkan untuk proses pendidikan dan penelitian.

Fitur yang ada dalam aplikasi *digital library* diantaranya selain berupa koleksi-koleksi *digital* yang dapat diakses oleh anggotanya, juga memiliki *link* ke koleksi perpustakaan *digital* milik perpustakaan digital lain yang tergabung dalam Indonesia Digital Library (IDLN). Untuk pendaftaran anggotanya tersedia fasilitas pendaftaran secara *online*. Aplikasi *digital library* ini dapat diakses pada alamat http://digilib.binadarma.ac.id.

Untuk mengakses koleksi *digital* yang dimiliki Universitas Bina Darma, masing-masing anggota memiliki *username* dan *password* khusus, yang dapat diperoleh dengan cara mendaftarkan diri sebagai anggotanya melalui fitur pendaftaran yang tersedia. *Administrator* akan memberikan konfirmasi keanggotaan melalui *email* yang didaftarkan. Dengan *username* dan *password* tersebut memungkinkan anggota untuk men*download* koleksi *digital* yang tersedia sesuai dengan kebijakan yang sudah ditentukan *administrator*. Masyarakat umum yang bukan anggota perpustakaan *Digital* Bina Darma hanya bisa melihat informasi koleksi, dan isi informasi koleksi yang terbatas yang biasanya hanya berupa deskripsi koleksi atau abstrak hasil penelitian, tugas akhir, thesis ataupun skripsi.

Web server aplikasi *Digital Library* Universitas Bina Darma dibangun menggunakan Apache 1.3.14 dengan sistem operasi Windows XP, bahasa pemrograman PHP versi 4.1.1, dan *database server* menggunakan MySQL versi 4. Tampilan halaman depan aplikasi *digital library* Universitas Bina Darma ditunjukkan ditunjukkan pada Gambar 2



Gambar 2. Tampilan *Digital Library* Universitas Bina Darma

3.1. Hasil Pengujian Keamanan *Digital Library* Universitas Bina Darma

Metode yang digunakan dalam penelitian ini adalah metode kualitatif dengan menggunakan beberapa *tools* berupa perangkat lunak dan cara-cara tertentu yang lazim digunakan untuk menguji keamanan aplikasi. Tahap-tahap yang dilakukan adalah sebagai berikut:

a. Tahap Inisiasi, pada tahap ini dilakukan penelusuran dan pengkajian literatur-literatur yang berhubungan dengan keamanan aplikasi.
b. Tahap Investigasi, pada tahap ini dilakukan penyelidikan terhadap *web server*, program aplikasi, dan *database server* yang digunakan.
c. Tahap Pengujian, pada tahap ini dilakukan pengujian terhadap keamanan aplikasi dengan menggunakan dua tools, yaitu *Acunetix Web Vurnerability Scanner* versi 6.5 (untuk menguji *web server* dan program aplikasi) dan *Shadow Database Scanner* versi 7.75 (untuk menguji *database server*) dengan metode yang lazim digunakan dalam pengujian keamanan aplikasi dan sistem.
d. Tahap Verifikasi, pada tahap ini dilakukan verifikasi terhadap keamanan aplikasi setelah dilakukan perbaikan-perbaikan atas dasar hasil investigasi dan pengujian pada aspek pemrograman maupun konfigurasi database server yang digunakan untuk memastikan bahwa database tersebut siap diterapkan untuk aplikasi *digital library*.

Aspek-aspek keamanan aplikasi yang diteliti meliputi :

a. *Web Server*
b. Program aplikasi
c. *Database Server*

Untuk melakukan analisis keamanan aplikasi aplikasi *Digital Library* Universitas Bina Darma, dilakukan menggunakan 2 (dua) *software*, yaitu :

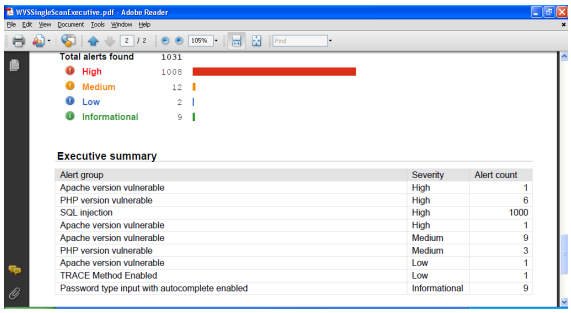
a. *Acunetix Web Vurnerability Scanner* versi 6.5
b. *Shadow Database Scanner* versi 7.75

3.1.1 Keamanan *Web Server*

Analisis keamanan pada sisi web server dalam penelitian ini dilakukan dengan menggunakan *software Acunetix Web Vurnerability Scanner* versi 6.5. Aspek-aspek yang dianalisis meliputi :

a. *Version Check*
b. *CGI Tester*
c. *Parameter Manipulation*
d. *Multirequest Parameter Manipulation*
e. *File Checks*
f. *Directory Checks*
g. *Web Applications*
h. *Text Search*
i. *File Uploads*
j. *Weak Passwords*
k. *Google Hacking Testing Database (GHDB)*
l. *Knowledge Base*
m. *Web Services-Parameter Manipulation*
n. *Web Services-Multirequest Parameter Manipulation*

Hasil analisis yang diperoleh menggunakan *software* tersebut adalah ditunjukkan pada Gambar 3 :



Gambar 3. Tampilan hasil Analisis *Web-Server*

Acunetix menetapkan skala 1 sampai 3 yang menyatakan tingkat *vulnerability* atas sistem yang di-scan. Berdasarkan hasil analisis di atas dapat diketahui bahwa tingkat ancaman terhadap *web server* situs *Digital Library* Universitas Bina Darma berada pada level 3 (*High*). Hal tersebut menunjukkan bahwa situs *Digital Library* Universitas Bina Darma memiliki banyak sekali celah yang memungkinkan terjadinya ancaman dan akses ilegal yang berpotensi merusak sistem. Diantaranya adalah sebagai berikut :

Tabel 1. Daftar Pesan Kesalahan Hasil Pemeriksaan Digital Library Universitas Bina Darma

No	Pesan (*Alert Summary*)	Peringatan (*Variations*)
1	*Apache Chunked-Encoding Memory Corruption Vulnerability:*	1
2	*PHP HTML Entity Encoder Heap Overflow Vulnerability*	1
3	*PHP multiple vulnerabilities*	1
4	*PHP POST file upload buffer overflow vulnerabilities*	1
5	*PHP unspecified remote arbitrary file upload vulnerability*	1
6	*PHP version older than 4.41*	1
7	*PHP Zend_Hash_Del_Key_Or_Index vulnerability*	1
8	*SQL Injection*	1000
9	*Unfiltered Header Injection in Apache 1.3.34/2.0.57/2.2.1*	1
10	*Apache Error Log Escape Sequence Injection Vulnerability*	1
11	*Apache version older than 1.3.27*	1
12	*Apache version older than 1.3.28*	1
13	*Apache version older than 1.3.29*	1
14	*Apache version older than 1.3.31*	1
15	*Apache version older than 1.3.34*	1
16	*Apache version older than 1.3.37*	1
17	*Apache version older than 1.3.39*	1
18	*Apache version older than 1.3.41*	1
19	*PHP mail function ASCII control character header spoofing vulnerability*	1
20	*PHP socket_iovec_alloc() integer overflow*	1
21	*PHP4 multiple vulnerabilities*	1
22	*Apache version up to 1.3.33 htpasswd local overflow*	1
23	*TRACE Method Enabled*	1
24	*Password type input with autocomplete enabled*	9
	Total Alerts Found	1031

Secara keseluruhan terdapat 1031 peringatan terhadap keamanan web server untuk situs digital library Universitas Bina Darma. Beberapa masalah yang terjadi terkait keamanan web server pada sistem digital library adalah sebagai berikut:

1. ***Apache Chunked-Encoding Memory Corruption Vulnerability*** : versi apache yang digunakan memungkinkan untuk dilakukannya serangan yang mengeksploitasi *buffer overflow* untuk serangan *Denial of Services (DOS).*
   Solusi yang disarankan adalah mengupgrade versi *Apache* ke versi yang lebih baru.
2. ***PHP HTML Entity Encoder Heap Overflow Vulnerability*** : merupakan celah keamanan PHP yang juga bisa dimanfaatkan untuk serangan DOS dan *remote code excecution*. Celah ini diakibatkan adanya boundary error dalam function htmlentities(), dan htmlspecialchars().
   Untuk mengatasinya bisa dilakukan dengan mengupgrade versi PHP dengan versi yang lebih baru.
3. ***PHP multiple vulnerabilities*** : merupakan celah PHP yang bisa dimanfaatkan untuk menjalankan code arbitary secara lokal maupun remote. Celah ini memanfaatkan keterbatasan privilege keamanan untuk mendapatkan informasi. Solusi yang disarankan untuk menutup celah ini juga dengan mengupgrade versi PHP terbaru.
4. ***PHP* POST *file upload buffer overflow vulnerabilities*** : yang juga merupakan celah di PHP yang bisa dimanfaatkan untuk menjalankan

code program penyerang. Solusi yang disarankan untuk celah ini adalah mengupgrade versi PHP.

5. ***PHP unspecified remote arbitrary file upload vulnerability*** : merupakan celah yang bisa dimanfaatkan penyusup untuk mengupload arbitary file. Untuk mengatasinya bisa dilakukan dengan mengupgrade versi PHP.
6. ***PHP* version *older than 4.41*** : seperti pada poin 1-5 versi PHP yang digunakan memiliki banyak celah yang bisa dimanfaatkan penyerang dalam melakukan *security bypass, cross site scripting, DOS* dan *system access.*
7. ***PHP Zend_Hash_Del_Key_Or_Index vulnerability*** : merupakan celah *script* aplikasi digital library Bina Darma yang bisa dimanfaatkan untuk menjalankan *code* program dan SQL *Injection*. Solusi yang bisa dilakukan adalah meng-*upgrade* versi PHP.
8. **SQL Injection** : merupakan celah yang paling banyak ditemui di aplikasi *digital library* Universitas Bina Darma. Celah ini memungkinkan penyerang menjalankan *query* SQL. Untuk itu perlu dilakukan perbaikan program aplikasi agar mem-*filter metacharacter* yang diinput oleh user. Terutama pada file /search.php.
9. ***Unfiltered Header Injection in Apache 1.3.34/2.0.57/2.2.1*** : merupakan kelemahan Apache yang *digunakan*, yang bisa dimanfaatkan penyerang untuk melakukan HTML injection (termasuk keode java script, VBScript, ActiveX, HTML atau Flash). Celah ini bisa diatasi dengan mengupgrade versi Apache terbaru.
10. ***Apache Error Log Escape Sequence Injection Vulnerability*** : celah ini bisa dimanfaatkan penyusup untuk membuat *file* dan atau menjalankan kode (dengan menggunakan terminal *emulator*). Solusinya dengan mengupgrade versi apache.
11. Untuk *point* no 11-18 (***Apache version older than 1.3.27- 1.3.41)***, seperti penjelasan poin-poin sebelumnya, versi *apache* yang digunakan di *server digital library* < versi 1.3.41, sehingga banyak celah keamanan yang mesti di *upgrade*.
12. ***PHP* mail *function ASCII control character header spoofing*** : Celah ini bisa dimanfaatkan penyerang untuk merubah *email header*. Solusinya diatasi dengan meng-*upgrade* versi PHP.
13. ***PHP socket_iovec_alloc() integer overflow*** : celah ini bisa dimanfaatkan penyerang untuk mengakibatkan *server crash* (DOS), dengan menjalankan *code arbitary* melalui argumen panjang *filename*. Celah ini juga bisa diatasi dengan meng-*upgrade* versi PHP.
14. ***PHP4* multiple *vulnerabilities*** : seperti penjelasan poin-poin sebelumnya versi PHP yang digunakan dalam *server digital library* ini memiliki banyak sekali celah keamanan yang bisa dimanfaatkan *hacker* untuk melakukan serangan DOS, sehingga perlu di *upgrade* versi PHP yang terbaru.
15. ***Apache version up to 1.3.33 htpasswd local overflow*** : celah versi Apache yang digunakan jika htpasswd tidak diseting menggunakan setuid (*id user*), untuk itu perlu dipastikan konfigurasi htpasswd menggunakan setuid dan tidak bisa diakses menggunkan kode CGI.
16. ***TRACE* Method *Enabled***, setting-an *HTTP TRACE method* di-*enable*. Ini bisa dimanfaatkan oleh penyerang untuk mendapatkan informasi header HTTP untuk memperoleh *cookies* dan data autentikasi. Untuk itu pada *web server* sebaiknya men-disable method ini.
17. ***Password Type Input with Autocomplete Enabled*** : ketika nama dan password dimasukkan ke form dan disubmit, browser akan menanyakan apakah user dan password akan disimpan. Dengan kelalaian pengguna maka hacker bisa memanfaatkan browser cache tersebut. Untuk itu seharusnya password autocomplete seharusnya di disable, dengan menggunakan kode seperti berikut <INPUT TYPE=”password” AUTOCOMPLETE=”off”>. Yaitu pada file-file berikut :
    a. */download.php*
    b. */login.php*
    c. */login_download.php*
    d. */registration.php*

3.1.2 Keamanan Program Aplikasi

Analisis keamanan pada sisi program aplikasi dalam penelitian ini dilakukan dengan menggunakan *software Acunetix web vurnerability scanner*. Aspek-aspek yang dianalisis meliputi:

a. *PHP Zend_Hash_Del_Key_Or_Index vulnerabilty.*
b. Celah keamanan *script* yang bisa dimanfaatkan untuk *SQL Injection.*
c. Celah keamanan *script* yang memungkinkan *Password type input with autocomplete enabled.*

Hasil analisis yang diperoleh menggunakan *software* tersebut telah tercakup dalam hasil analisis sebelumnya.

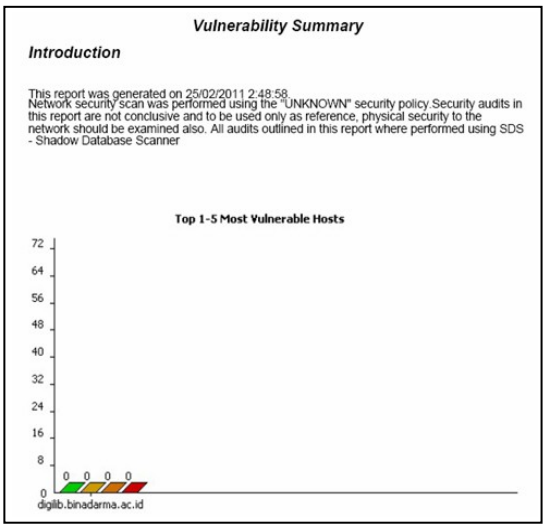
3.1.3 Keamanan *Database Server*

Analisis keamanan pada sisi *database server* dalam penelitian ini dilakukan dengan menggunakan *software Shadow Database Scanner* versi 7.75. Aspek-aspek yang dianalisis meliputi :

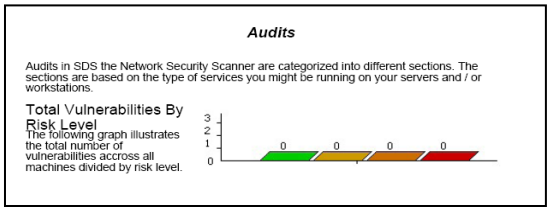
1. *Audit*, meliputi :
    a. *IP address*
    b. *Host name*
    c. *Average ping response*
    d. *TCP port*
2. *Vulnerability*

Tampilan hasil analisis pada situs *digital library* Universitas Bina Darma secara berturut-turut untuk *Summary* ditunjukkan pada Gambar 4, *Audit*

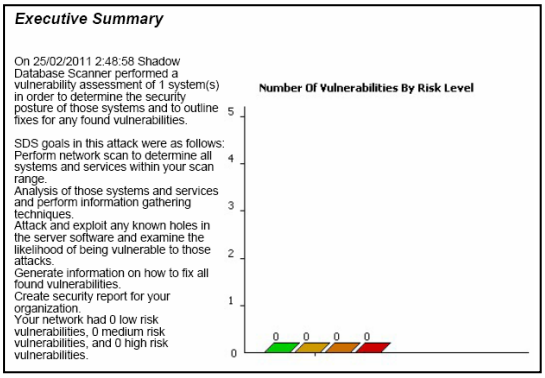
ditunjukkan pada Gambar 5, dan *Vulnerability* ditunjukkan pada Gambar 6.



Gambar 4. Tampilan hasil analisis *Database Server-Summary*



Gambar 5. Tampilan hasil analisis *Database Server-Audit*



Gambar 6. Tampilan hasil analisis *Database Server-Vulnerability*

Berdasarkan hasil analisis di atas maka dapat diketahui bahwa keamanan *database server* untuk situs *digital library* Universitas Bina Darma dapat dinyatakan aman terhadap kemungkinan adanya ancaman dan akses ilegal yang berpotensi merusak.

3.1.4 Verifikasi Keamanan *Digital - Library*

Dari hasil pengujian keamanan yang dilakukan pada *Server Digital Library* Universitas Bina Darma, maka secara umum ada 4 celah yang bisa dimanfaatkan oleh penyusup untuk melakukan serangan dengan menggunakan teknik *Cross-Site Scripting* dan *SQL Injection*, yaitu :

a. Versi *web server digital library* Universitas Bina Darma yang masih menggunakan versi 1.3.14, yang memiliki banyak celah kerawanan. (lihat tabel 4.1 dengan no urut pesan 1, 10 s.d. 18).
b. Versi PHP yang digunakan yaitu versi 4.1.1 yang memiliki banyak celah untuk dilakukan *SQL Function*. (lihat tabel 4.1 dengan no urut pesan 2 s.d. 7, 19 s.d. 21).
c. Kelemahan *script* aplikasi yang memungkinkan dilakukan *SQL Injection*.
d. Kelemahan konfigurasi yang bisa dimanfaatkan penyusup.

Untuk itu ada 4 hal yang harus dilakukan untuk memperbaiki celah keamanan tersebut, yaitu :

a. Melakukan *update* versi *web server* Apache
b. Melakukan *update* versi PHP
c. Melakukan perbaikan *scipt*
d. Memeriksa ulang konfigurasi *server*.

3.2 Perbaikan Celah Keamanan

3.2.1 *Upgrade* Sistem *Digital Library* Universitas Bina Darma

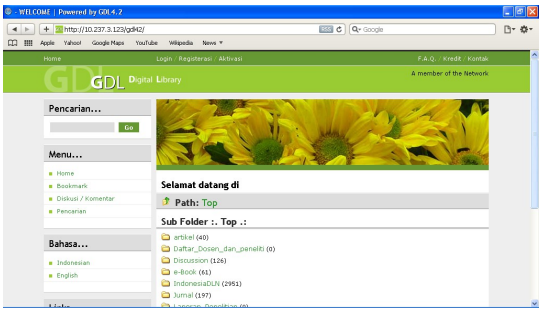
Digital Library Universitas Bina Darma dikembangkan menggunakan GDLN versi 4.0. GDLN sendiri adalah *software open source* yang dikembangkan oleh IDLN khusus untuk perpustakaan digital di Indonesia. Saat ini GDLN telah mengeluarkan versi 4.2 yang diluncurkan tahun 2006. GDLN versi 4.2 ini merupakan perbaikan dari versi-versi sebelumnya. Untuk itu dalam penelitian ini, untuk memperbaiki celah keamanan di *digital library* Universitas Bina Darma, maka dilakukan pengujian dengan menggunakan GDLN versi 4.2.

Adapun tahapan yang dilakukan adalah sebagai berikut :

a. Menyiapkan *server* uji coba, dengan konfigurasi sebagai berikut: Sistem Operasi Linux Ubuntu 10.04, Apache versi 2.2.14, PHP versi 5.3.2 dan MySQL versi 5.1.4.1.
b. Instalasi GDLN versi 4.2.
c. Melakukan migrasi *folder*, *metadata*, relasi, *file*, pengguna, dan *log* akses.
d. Pengujian celah keamanan.

Tampilan menu migrasi dan halaman utama pada situs *digital library* Universitas Bina Darma versi 4.2 secara berturut-turut untuk menu Migrasi ditunjukkan pada Gambar 7, halaman utama ditunjukkan pada Gambar 8.

Gambar 7. Migrasi GDL 4.0 ke 4.2



Gambar 8. *Digital Library upgrade* ke GDL 4.2

3.2.2 Evaluasi Celah Keamanan Sistem *Digital Library* yang di *Upgrade*

Setelah melakukan proses *upgrade* maka dilakukan kembali proses uji coba dengan menggunakan tools *Acunetix Web Vulnerabilities* dan *Shadow Data Scanner* untuk mencari celah keamanan sistem yang telah di-*upgrade*. Aspek yang dilihat juga berupa aspek keamanan *web server*, *aplikasi* dan *database*. Hasil Analisis bisa dilihat pada gambar 9.

Gambar 9. Hasil Web Analisis Sistem *Digital Library* yang sudah di update

Dari hasil pengujian yang dilakukan setelah sistem digital library diupdate menunjukkan tingkat keamanan sistem berada pada level 1(*low*). Dengan rincian sebagai berikut :

Tabel 2. Daftar Pesan Kesalahan Hasil Pemeriksaan *Digital Library* Universitas Bina Darma versi GDL 4.2

No	Pesan (*Alert Summary*)	Peringatan (*Variations*)
1	*User credentials are sent in clear text*	6
2	*Bonjour service running*	1
3	*SMB list shares*	1
4	*SMB null session*	1
	Total Alerts Found	9

Hal ini menunjukkan setelah di *upgrade* maka *system digital library* Universitas Bina Darma cukup aman. Adapun peringatan-peringatan yang ada dapat dijelaskan sebagai berikut :

a. ***User credentials are sent in clear text*** ; dikarenakan *protocol web* yang digunakan menggunakan *protocol* http (*port* 80), semua aplikasi *web* yang menggunakan *protocol* http maka pada saat data ditransmisi maka data tidak di enkripsi. Celah ini bisa dimanfaatkan penyerang yang melakukan *sniffing* di jaringan, karena data yang ditangkap berbentuk *clear text*. Untuk mengatasinya bisa saja di implentasikan *digital library* Univrsitas Bina Darma menggunakan *protocol* https (port 443), sehingga *link*-nya menjadi https://digilib.binadarma.ac.id
b. Sedangkan untuk poin 2, 3 dan 4 yaitu ***Bonjour service running***, dan ***SMB list shares*** sebenarnya bukan kelemahan sistem *digital library* Universitas Bina Darma. Pesan ini muncul terkait dengan aplikasi pada PC yang digunakan saat pengujian membuka *port* UDP 5353 untuk konfigurasi jaringan *user* dan TCP Port 445 untuk *sharing printer*. Port-port tersebut pada implementasi di *server real* tidak dibuka.

Jadi dari hasil yang didapatkan menunjukan celah keamanan *digital library* Universitas Bina Darma yang telah di *update* sangat baik.

3.2.3 Evaluasi Celah Keamanan *Database System*

Setelah melakukan proses *upgrade* maka dilakukan kembali proses uji coba analisis keamanan pada sisi *database server* dengan menggunakan *software Shadow Database Scanner* versi 7.75. Aspek-aspek yang dianalisis sama dengan bahasan sebelumnya di bagian 3.1.3, yaitu meliputi :

a. *Audit*, meliputi (*IP address, Host name, Average ping response, TCP port*).
b. *Vulnerability*

Tampilan hasil analisis pada situs *digital library* Universitas Bina Darma secara berturut-turut untuk *Summary*, *Audit* dan *Vulnerability* sama dengan bahasan sebelumnya di bagian 3.1.3.

Berdasarkan hasil analisis di atas maka dapat diketahui bahwa keamanan *database server* setelah di *upgrade* untuk situs *digital library* Universitas Bina Darma dapat dinyatakan aman terhadap kemungkinan adanya ancaman dan akses *ilegal* yang berpotensi merusak.

4. Kesimpulan

Dari hasil Penelitian Analisis *Web Vulnerabilites* untuk meningkatkan keamanan *website* studi kasus *digital library* Universitas Bina Darma dapat disimpulkan sebagai berikut:

a. Penelitian ini berhasil melakukan analisis terhadap aspek-aspek keamanan sistem yang meliputi keamanan *web server*, program aplikasi dan *database server* pada *database* yang

digunakan pada sistem *digital library* Universitas Bina Darma.

b. Berdasarkan hasil analisis diketahui bahwa tingkat ancaman terhadap *web server* aplikasi *digital library* Universitas Bina Darma berada pada level 3 (*High*). Hal tersebut menunjukkan bahwa pada aplikasi *digital library* Universitas Bna Darma masih banyak sekali terdapat celah-celah yang memungkinkan terjadinya ancaman dan akses *ilegal* yang berpotensi merusak sistem. Sedangkan keamanan *database server* untuk aplikasi *digital library* Universitas Bina Darma aman terhadap kemungkinan adanya ancaman dan akses *ilegal* yang berpotensi merusak.
c. Secara umum terdapat 4 celah keamanan pada sistem digital library Universitas Bina Darma, yaitu versi web server Apache dan PHP yang tidak update, kelemahan script aplikasi dan kelemahan konfigurasi. Celah ini bisa dimanfaatkan penyerang untuk melakukan Cross-Site Scipting dan SQL Injection.
d. Dengan perbaikan sistem digital library Universitas Bina Darma dengan meng-*update web server* Apache, PHP dan versi GDLN maka didapatkan sistem dengan keamanan yang cukup baik..

Adapun saran yang dapat diberikan dari hasil penelitian tentang Analisis *Web Vulnerabilites* untuk meningkatkan keamanan *website* studi kasus *digital library* Universitas Bina Darma sebagai berikut :

a. Untuk dapat meningkatkan keamanan *website digital library* ini setiap waktu, hal yang perlu dilakukan adalah harus senantiasa melakukan *upgrade* sistem untuk *web server*, program aplikasi dan *database server* pada *database* yang digunakan pada sistem *digital library* Universitas Bina Darma.
b. *Port* yang diaktifkan pada komputer *server digital library* adalah *port* yang hanya digunakan dan diperlukan saja. Untuk yang tidak terpakai sebaiknya di non aktifkan saja, untuk mencegah pintu masuk bagi *hacker* untuk melakukan *hacking* maupun penetrasi ke jaringan komputer kita.

Daftar Pustaka

[1] Auronen, Lauri. 2002. *Tool-Based Approach to Assessing Web Application Security.* Helsinki University of Technology Telecommunications Software and Multimedia Laboratory. { Lauri.Auronen@hut.fi}

[2] Bhasin, Shweta with NIIT. 2003. *Web Security Basic.* Cincinnati, Ohio, United State of America. Premier Press, a division of Course Technology.

[3] Chandrax. July 31st, 2008. *Action Research / Penelitian Tindakan*. Posted in Pustaka. <http://chandrax.net76.net/?p=7>, diakses 21 Oktober 2010,

[4] Fredrik Valeur, Darren Mutz, dan Giovanni Vigna. 2006. *A Learning-Based Approach to the Detection of SQL Attacks.* University of California, Santa Barbara. Reliable Software Group Department of Computer Science. {fredrik,dhm,vigna}@cs.ucsb.edu.

[5] GAUR, NALNEESH. 2000. *Assessing the security of your web applications*; *Linux Journal* 2000(72es):3. ISSN 1075-3583.

[6] Philipp Vogt, Florian Nentwich, Nenad Jovanovic, Engin Kirda, Christopher Kruegel, dan Giovanni Vigna. 2006. *Cross-Site Scripting Prevention with Dynamic Data Tainting and Static Analysis.* Austria ; Secure Systems Lab Technical University Vienna. Santa Barbara - USA ; University of California. {pvogt, fnentwich, enji, ek, chris}@seclab.tuwien.ac.at. vigna@cs.ucsb.edu.

[7] Rahardjo, Budi. 2002. *Keamanan Sistem Informasi Berbasis Internet*. Bandung - PT Insan Infonesia dan Jakarta - PT. INDOCISC. {rahardjo@insan.co.id}.

[8] Riadi, Imam. Jazi Eko Istiyanto. 2003. *Analisis Kelemahan Cross Site Scripting pada PHP Nuke untuk Keamanan Website*. Yogyakarta. Fakultas MIPA Universitas Ahmad Dahlan, Fakultas MIPA Universitas Gadjah Mada. {imam_riadi@softhome.net, jazi@ugm.ac.id}.

[9] Sunyoto, Andi. 2003. *Metode Penyerangan Website Menggunakan SQL Injection.* Yogyakarta. STMIK Amikom.

[10] Ter Louw, Mike. V.N. Venkatakrishnan. 2009. *BLUEPRINT: Robust Prevention of Cross-site Scripting Attacks for Existing Browsers.* Chicago, USA. Department of Computer Science University of Illinois. {mter, venkat} @cs.uic.edu.