Jurnal Teknologi Informasi dan Ilmu Komputer (JTIIK) DOI: 10.25126/jtiik.202071928
Vol. 7 No. 4, Agustus 2020, hlm. 853-860 p-ISSN: 2355-7699
Akreditasi KEMENRISTEKDIKTI, No. 30/E/KPT/2018 e-ISSN: 2528-6579

ANALISIS KEAMANAN *WEBSITE OPEN JOURNAL SYSTEM* MENGGUNAKAN METODE *VULNERABILITY ASSESSMENT*

Imam Riadi[1], Anton Yudhana[2] Yunanri.W*[3]

[1]Program Studi Sistem Informasi, Universitas Ahmad Dahlan, Indonesia
[2]Program Studi Teknik Elektro, Universitas Ahmad Dahlan, Indonesia
[3]Program Studi Teknik Informatika, Universitas Ahmad Dahlan, Indonesia
Email: [1]imamriadi@is.uad.ac.id, [2]eyudhana@ee.uad.ac.id [3]yunanriw1607048008@webmail.uad.ac.id
*Penulis Korspondensi

(Naskah masuk: 11 April 2019 diterima untuk diterbitkan: 5 Juli 2019)

Abstrak

Open Journal System (OJS) merupakan perangkat lunak yang berfungsi sebagai sarana publikasi ilmiah dan digunakan diseluruh dunia. OJS yang tidak dipantau beresiko diserang oleh *hacker*. Kerentanan yang di timbulkan oleh *hacker* akan berakibat buruk terhadap performa dari sebuah OJS. Permasalahan yang dihadapi pada sistem OJS meliputi *network*, *port discover*, proses audit *exploit* sistem OJS. Proses audit sistem pada OJS mencakup *SQL Injection*, melewati *firewall* pembobolan *password*. Parameter input yang digunakan adalah IP *address* dan *port open access*. Metode yang digunakan adalah *vulnerability assessment*. Yang terdiri dari beberapa tahapan seperti *information gathering* atau *footprinting*, *scanning vulnerability*, *reporting*. Kegiatan ini bertujuan untuk mengidentifikasi celah keamanan pada *website open journal system* (OJS). Penelitian ini menggunakan *open web application security project* (OWASP). Pengujian yang telah dilakukan berhasil mengidentifikasi 70 kerentanan *high*, 1929 *medium*, 4050 *low* pada OJS, Total nilai *vulnerability* pada OJS yang di uji coba sebesar 6049. Hasil pengujian yang dilakukan menunjukkan bahwa pada OJS versi 2.4.7 memiliki banyak celah kerentanan atau *vulnerability*, tidak di rekomendasi untuk digunakan. Gunakanlah versi terbaru yang dikeluarkan oleh pihak OJS *Public knowledge project* (PKP).

Kata kunci: *OJS, Security, Vulnerability Assessment, OWASP*.

SECURITY ANALYSIS OPEN JOURNAL SYSTEM WEBSITE USING VULNERABILITY ASSESSMENT METHOD

Abstract

The Open Journal System (OJS) is A software that functions as a means of scientific publication and is used throughout the world. OJS that is not monitored is at risk of being attacked by hackers. Vulnerabilities caused by hackers will adversely affect the performance of an OJS. The problems faced by the OJS system include the network, port discover, OJS system audit exploit process. The system audit process on the OJS includes SQL Injection, bypassing the firewall breaking passwords. The input parameters used are the IP address and open access port. The method used is a vulnerability assessment. Which consists of several stages such as information gathering or footprinting, scanning vulnerability, reporting. This activity aims to identify security holes on the open journal system (OJS) website. This study uses an open web application security project (OWASP). Tests that have been carried out successfully identified 70 vulnerabilities high, 1929 medium, 4050 low in OJS, the total value of vulnerability in OJS which was tested was 6049. The results of tests conducted showed that in OJS version 2.4.7 had many vulnerabilities or vulnerabilities, not on recommendations for use. Use the latest version issued by the OJS Public Knowledge Project (PKP).

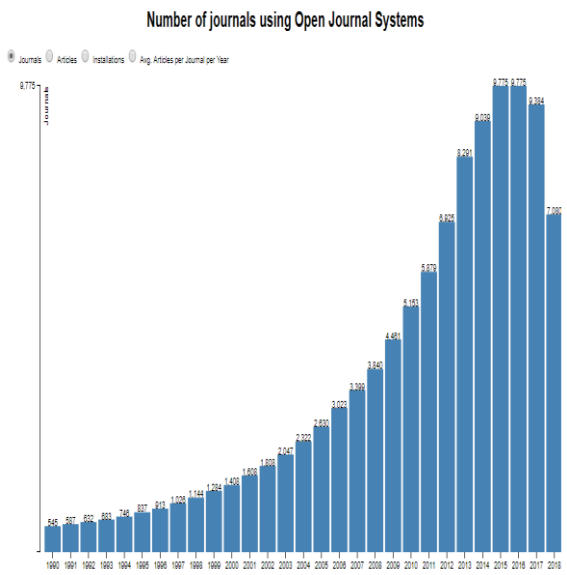
Keywords: *OJS, Security, Vulnerability Assessment, OWASP*.

1. PENDAHULUAN

Open Journal System (OJS) merupakan perangkat lunak atau *software* yang bersifat terbuka atau *open-source* (Manalu dkk, 2017). OJS bertujuan sebagai sarana publikasi karya ilmiah. OJS merupakan salah satu aplikasi *web* yang cukup banyak digunakan. Penggunaan OJS dari tahun 1990 sampai tahun 2018 mengalami peningkatan penggunaan semakin tinggi. Seperti pada Gambar 1.

Penggunaan OJS yang semakin banyak, tidak menutup kemungkinan akan adanya serangan *hacker* yang dapat megganggu, mulai dari kurangnya

performa OJS *error* akses, bahkan dapat diambil alih oleh *hacker*. Keamanan, menjadi solusi pada sistem OJS. Keamanan sistem OJS diperlukan untuk melindungi informasi yang terdapat didalamnya. Keamanan yang dimaksud mengacu pada kerahasiaan data, integritas dan ketersediaan layanan pada sistem yang diterapkan.



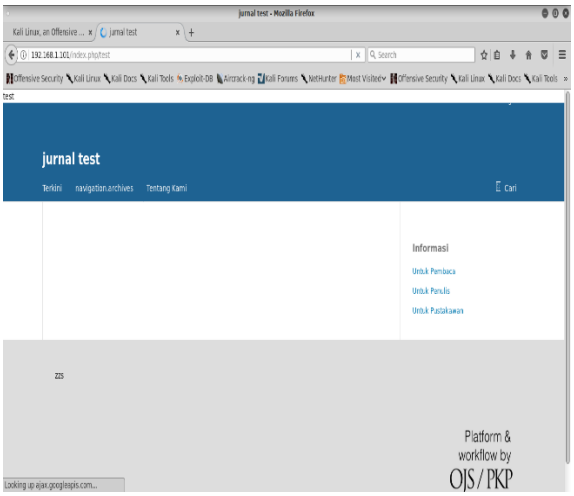
Gambar 1. Menunjukkan penggunaan OJS semakin meningkat dari tahun 1990 sampai tahun 2018.

Sebuah Karya yang diterbitkan pada *Open Access* dapat dilihat, dibaca dan digunakan oleh semua orang yang tertarik, sehingga memungkinkan penelitian akademik memiliki dampak yang lebih besar pada dunia. Kemudahan akses pada sistem Jurnal ada di *platform* yang menjanjikan diseluruh dunia (Akhoon dkk, 2018).

Adanya aset data penting berupa informasi sebuah organisasi yang perlu dilindungi dengan mengikuti pendekatan yang komprehensif dan terstruktur terhadap risiko organisasi yang mungkin dihadapi. Masalah keamanan dibutuhkan penerapan metode yang dapat menjamin keamanan data, transaksi, dan komunikasi. Tidak adanya keamanan pada sistem akan berdampak buruk. *Hacker* dengan mudah dapat mengambil alih sistem yang dibangun. Hal ini menimbulkan permasalahan pada data yang bersifat pribadi, maupun data yang sangat penting sebuah perusahaan atau lembaga yang seharusnya tidak diketahui oleh orang lain, akan tetapi dapat di akses oleh *hecker*. *Hacker* merupakan seseorang yang memiliki kemampuan tinggi dibidang teknologi informasi.

Perlunya mengambil langkah cepat untuk mengamankan OJS dan jika diabaikan maka *website* OJS yang dimiliki oleh suatu badan institusi baik milik perintah, swasta, maupun perseorangan dapat mengalami kerugian yang diakibatkan oleh para *hacker*.

Dalam studi kasus keamanan, OJS bertujuan untuk mencari celah keamanan atau *vulnerability*, untuk segara diperbaiki. Kerentanan atau *vulnerability* pada aplikasi OJS dapat mengganggu performa OJS, bahkan dapat di ambil alih oleh *hacker*(Mutemwa dkk, 2018). Contoh OJS yang ditampilkan pada Gambar 2.



Gambar 2. Pengujian pada *Open Journal System* (OJS), setelah di *install* pada *server*.

Tool yang digunakan dalam melakukan audit sistem OJS menggunakan *Open Web Application Security Project* (OWASP). *Tool* yang digunakan memiliki kemampuan untuk mendeteksi beberapa celah kerentanan antara lain :

- Serangan *Injection*
- Serangan *Broken authentication and session management*
- Serangan *Cross site scripting* (*XSS*)
- *Insecure direct object references*
- *Security misconfiguration*
- *Sensitive data exposure*
- *Missing function level access control*
- *Cross site request forgery (CSRF)*
- *Using Components with known vulnerabilities*
- *Unvalidated redirects and forwards*

2. METODE PENELITIAN

Objek penelitian ini mencari *vulnerability* atau Celah kerentanan pada file sistem, *webserver open journal system* (OJS). Objek penelitian ini adalah OJS mencakup celah keamanan atau *vulnerability* pada sistem OJS yang berjalan pada *website* yang di kelola oleh *admin*, baik organisasi, bada atau lembaga pendidikan yang telah tersertivikasi oleh dunia (Bo Wang dkk, 2018).

Metode yang digunakan pada penelitian ini adalah metode *penetration testing* yang berfokus pada *vulnerability assessment* (Gorbenko dkk, 2017). BlackBox adalah merupakan jenis pengujian sistem tanpa mengetahui struktur rancang bangun pada sebuah sistem. Pengujian dilakukan pada *open*

journal system (OJS) dan informasi mengenai jaringan maupun informasi lainnya harus di cari sendiri oleh penguji, cara ini merupakan cara yang menghasbiskan banyak waktu serta biaya yang besar(Kuniawan dkk, 2018).

Adapun sekenario pengujian dan analisis sistem OJS adalah sebagai berikut:

a. Skenario penyerangan mengacu pada *framework* yang *powerful* seperti OWASP *tool*. Ditemukan nya *vulnerability*, pada list pada target yang di perbaiki agar mudah untuk diperbaiki dari hasil exploitasi oleh *tool* OWASP (Yunanri dkk, 2018).
b. Penyerangan pada *page admin* dan *password* tidak akan dirubah pada saat *penetration guide* dalam mencari *Error code* (Letian dkk, 2014).
c. *Listing direktori class java* mengarah pada memperoleh kode aplikasi yang mengandung kerentanan atau celah pada sistem OJS.
d. Konfigurasi *stack trace* ke *user* bertujuan mencari celah yang memiliki potensial cacat pada *application* OJS.

Gambar 3. *Flowchart* Alur Pengujian Sistem *Vulnerability Assessment* pada OJS

Berdasarkan Gambar 3. langkah awal pengujian penelitian perlu dilakukan pengumpulan informasi. Mengenai posisi fisik *server*, jenis jaringan yang digunakan, perangkat yang digunakan dan berbagai informasi yang berkaitan dengan *Webserver* OJS yang bisa didapatkan dengan aplikasi antara lain:

- *Network discover* bertujuan mendapatkan informasi memindai pada *port-port host* yang terbuka.
- *Vulnerability scanning* bertujuan mencari celah kerentanan atau *vulnerability* pada *website*, *server* dan lain-lain.
- *Result analysis* merupakan kesimpulan akhir, berupa tabel dari jumlah nilai dari sebuah penelitian yang telah dilakukan.

Kemudian dilakukan pemindaian celah keamanan atau *vulnerability*. Yang ada pada sistem dengan menggunakan aplikasi OWASP(Ghanem dkk, 2013).

2.1 Sistem Analisis

Analisis sistem dilakukan untuk memperoleh informasi dari sistem yang bertujuan untuk melakukan analisis kelemahan sistem. Tahapan-tahapan yang dilakukan dalam mengumpulkan informasi adalah dengan menggunakan modul CEH (Elizabeth dan Jimenez, 2016). Beberapa alat yang diimplementasikan, yaitu :

a. *Footprinting and Network Discover*.

Fase ini adalah menemukan struktur rancang bangun dari keamanan jaringan pada target sasaran yang dituju sebagai barometer metodologi:

Whois adalah suatu prosedur untuk mendapatkan informasi mengenai sebuah *domain*, alamat, No.telpon, alamat *email*, kapan domain ini di daftarkan dan kapan domain ini akan kadarluarsa.

Nslookup adalah *tool* berguna untuk mengetahui IP dari sebuah domain. Disamping itu juga dapat berguna untuk mendiagnosa permasalahan jaringan yang berkaitan dengan *DNS*.

Scanning port adalah sebuah prosedur aplikasi yang dirancang untuk menyelidiki *server* atau *host port* terbuka. Aplikasi sering digunakan oleh *administrator* untuk memverifikasi keamanan jaringan.

HttpRecon adalah sebuah prosedur untuk mengumpulkan informasi pada *network*, *webserver*, yang bersifat *hypertext transfer protokol*.

b. *Scanning Vulnerability* :

Tujuan *scanning vulnerability* adalah mencari celah akamanan yang terdapat pada target mencapkup beberapa seperti SQL *Injection*, *Cross Site Scripting* (XSS), *Remote OS Command*, *Path Transversal*, *Private IP Disclosure*. pada suatu sistem operasi atau aplikasi (Kurniawan dkk, 2017).

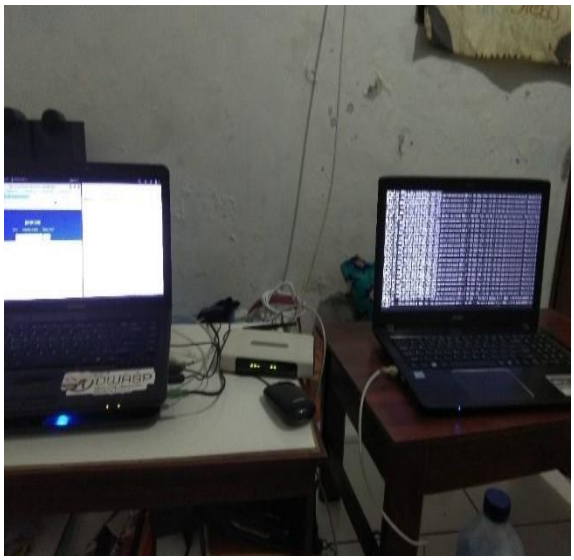
c. *Reporting* :

Merupakan laporan dari awal langkah sampai akhir berbentuk *file document* sebagai rekomendasi langkah-langkah pencegahan perbaikan pada sistem baik perusahaan, lembaga pendidikan dan organisasi(Riadi. I dan Umar. R, 2016).

3. HASIL DAN ANALISIS

Simulasi kasus dilakukan kegiatan percobaan melakukan Audit OJS pada *server* OJS yang dibangun sendiri menggunakan *tool* OWASP.

Kegiatan yang dilakukan antara lain melakukan audit berupa *requsting* pada *server* yang sudah terintergrasi *open jurnal system* (Barghutihi dkk, 2018). Seperti pada Gambar 4.



Gambar 4. Simulasi Audit *Open Journal System* (OJS) pada *Server*.

3.1. *Footprinting* atau *Information Gathering*.

- *Whois.*

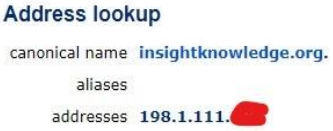
Merupakan suatu prosedur untuk mendapatkan informasi mengenai sebuah *domain*, alamat, No.telpon, alamat *email*, kapan *domain* ini di daftarkan dan kapan *domain* ini akan kadarluarsa.

Gambar 5. Simulasi Audit *Open Journal System* pada *Server* OJS.

Gambar 5. Merupakan hasil *scanning* dari *tool Central OPS* secara *real-time*, pada OJS versi 2.4.7. Hasil *scanning* oleh *tool Central OPS* menampilkan informasi (Naik dan Jenkins, 2018).

- *Nslookup*.

Nslookup bertujuan untuk mengetahui IP dari sebuah *domain*. Disamping itu juga dapat berguna untuk mendiagnosa permasalahan jaringan yang berkaitan dengan *DNS*.

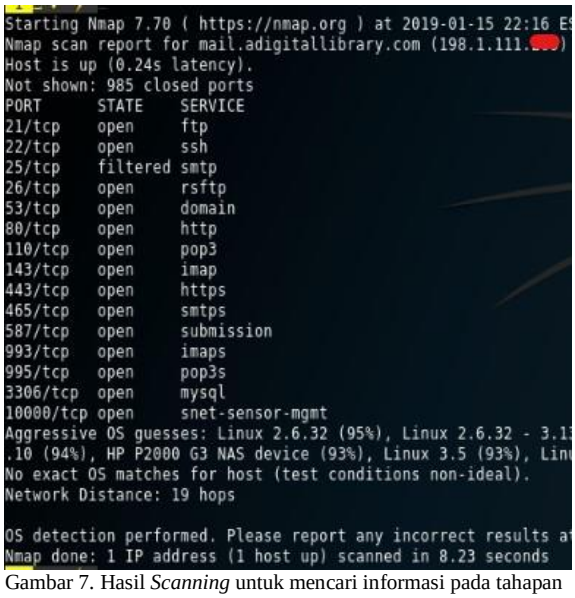


Gambar 6. Menampilkan alamat *Website* dan IP *Address Open Journal System* pada *Server* OJS.

Gambar 6. merupakan hasil *scanning* dari *tool Central OPS* secara *real-time*, pada OJS versi 2.4.7. hasil *scanning* oleh *tool Central OPS* menampilkan informasi berupa " IP *Address*".

- *Scanning Port Discover*

Merupakan sebuah prosedur aplikasi yang dirancang untuk menyelidiki *server* atau *host port* terbuka. Aplikasi sering digunakan oleh *administrator* untuk memverifikasi keamanan jaringan.



Gambar 7. Hasil *Scanning* untuk mencari informasi pada tahapan *Information Gathering*.

Gambar 7. Mendeteksi beberapa *port* yang terbuka akan sangat berbahaya, karena dengan adanya celah ini *hacker* dengan mudah akan masuk (Kurniawan, Riadi.I, 2018) (Yudhana dkk, 2018).

- *HttpRecon*

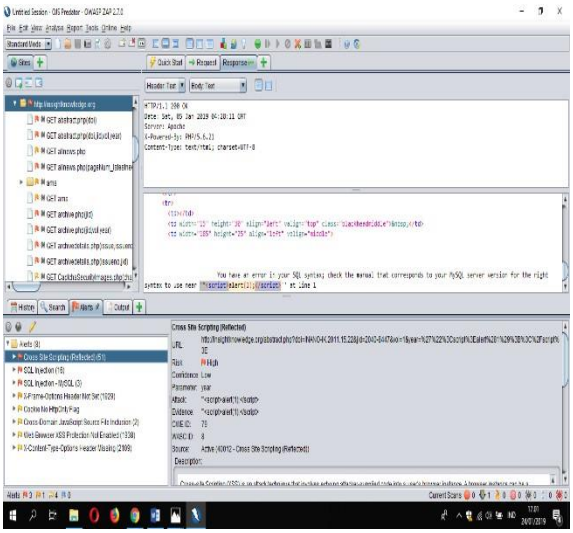
Sebuah prosedur untuk mengumpulkan informasi pada *network*, *webserver*, yang bersifat *hypertext transfer protokol*.

Gambar 8. Hasil *Scanning* untuk mencari informasi pada tahapan *Information Gathering*.

Gambar 8. Hasil *scanning* oleh *tool Http Recon* berupa *website* yang dapat melakukan audit secara otomatis yang berupa *website* Pada OJS versi 2.4.7. Hasil *scanning Http Recon* menampilkan informasi berupa " *noktifikasi place chek your Query* and *tray again*" yang menjelaskan adanya kesalahan pada *query* pada *file* sistem OJS sub *Querynya* (Firdausy dkk, 2008).

3.2. *Scanning Vulnerability* (Memindai Celah keamanan).

Vulnerability scanning bertujuan mencari celah kerentanan atau *vulnerability* pada *website, server* dan lain-lain.



Gambar 9. Hasil *scanning* oleh OWASP, diperoleh 8 (sepuluh) sub file sistem yang terindikasi memiliki *vulnerability*.

3 (tiga) *alert* tersebut menampilkan adanya 3 bagian celah kerentanan atau *vulnerability* (Han dkk, 2018)(Mutemwa dkk, 2018)(Un, dkk, 2018), diantranya : 70 *high risk*, 1929 *medium risk*, 4049 *low risk*, menghasilkan *alert* di antaranya:
- *Cross Site Scripting*.
- SQl *Injection*,
- SQL *Intjection MySQL*.
- *X-frame-Options Header Not Set*.
- *Cookie No. HttpOnly flag.*
- *Cross Domain Javascript Source file Inclusion*.
- *Web Browser XSS Protection not Enabled.*
- *X-content-type-options header missing*.

Seperti yang ditampilkan pada Gambar 9.

3.3. Hasil Analisis atau *Result Analysis*

Tabel 1. Pengujian yang telah dilakukan pada *Open Journal System* (OJS) mendeteksi 8 sub file *vulnerability,high, medium, low*.

No	Alert	Risk			Ket.
		High	Medium	Low	
1	Cross Site Scripting	51			Hasil dari Audit ini menunjukkan bahwa 3 file sub system perlu penanganan sesegera mungkin, pada sub file sistem yang memiliki High Risk Berjumlah 70 file
2	SQL Injection	16			
3	SQL Injection - My SQL	3			
4	X-Frame-Options Header not Set		1929		Untuk Medium Risk pada OJS ini tahap mengkhawaturkan harus segera untuk diperbaikan oleh admin, pengelola OJS. Berjumlah 1929
5	Cookie No. Http Only Flag			1	Sementara pada Posisis Low Risk masih berada pada keadaan kerusakan ringan, Berjumlah 4050
6	Cross Domain Java Script Source fikle Inclusion			2	
7	Web Browser XSS Peotection not Enable			1938	
8	X-Content-Type Options Header Missing			2109	
	Total Vulnerability		6049		

Sumber: hasil *scanning* mendeteksi 8 *vulnerability* secara keseluruhan pada OJS, menggunakan *tool* OWASP.

3.4. Rekomendasi untuk perbaikan sistem OJS atau *Countermeasure*.

Tabel 2. *Countermeasure* merupakan sebuah saran yang direkomendasi oleh *tool* OWASP yang memliki standar dan kualitas tinggi pada bidang *IT Security* (Han et al, 2018). Rekomendasi unttuk ditindaklanjuti seperti pada tabel dibawah ini:

Tabel 2. Rancangan Analisis Komputasi

No.	Nama Sub file sistem *vulenerability*	Jumlah *vulnerability*	Rekomendasi perbaikan (*Countermeasure*)
1.	*Cross Site Scripting* (*XSS*)	51	Tinjauan kode sumber halaman kesalahan kustom, pertimbangkan untuk menerapkan mekanisme untuk memberikan refrensi atau pengenalan kesalahan yang memiliki unsur yang unik pada klien (*browser*) pada *server* dan tidak

No.	Nama Sub file sistem *vulenerability*	Jumlah *vulnera bility*	Rekomendasi perbaikan (*Countermeasure*)
			untuk dipaparkan (A.Kurniawandkk, 2017).
2.	*SQL Injection*	16	Jangan mudah percaya bagi yang menginput terlebih oleh klien bahkan jika ada validasi yang mengatasnakan klien, secara umum ketika pemeriksaan pada *server*. Jika apliksi menggunakan *JDBC*, disarankan menggunakan preparedstatement atau *callablestatement* dengan parameter dilewati. Jika aplikasi menggunakan *ASP*, gunakan objeck perintah *ADO* dengan pemeriksaan tipe yang kuat dan jangan gunakan * tidak digabung dengan *string* kedalam *query* pada suatu prosedur atau menggunakan '*exec*', '*exec direct'*., serta fungsionalitas *query* SQL dinamis pada skup string (FIRDAUSY, SADRI, 2008).
3.	*SQL Injection-MySQL Injection*	3	Jika Prosedur yang Disimpan dalam *database* dapat digunakan, gunakanlah.Jangan * tidak menggabungkan *string* ke dalam *query* dalam prosedur tersimpan, atau gunakan *'exec'*, *'exec direct'*, atau fungsionalitas yang setara!Jangan membuat *query SQL* dinamis menggunakan penggabungan *string* sederhana. Kabur dari semua data yang diterima dari klien.Terapkan 'daftar putih' karakter yang diizinkan atau 'daftar hitam' karakter yang dilarang di inputpengguna. Menerapkan prinsip *privilege* terkecil dengan menggunakan pengguna basis data yang paling *privilege*. Secara khusus, hindari menggunakan pengguna basis data 'a' atau *'db-owner'*. tidak menghilangkan injeksi SQL, tetapi meminimalkan dampaknya. Berikan akses basis data minimum yang diperlukan untuk aplikasi (E. Kurniawan 2018)
4.	*X-Frame -Options Header not set*	1929	Pastikan filter XSS *browser Web* diaktifkan, dengan mengatur *header respons* X-*XSS-Protection HTTP* ke '1'.
5.	*Cookie No. HttpOnly Flag*	1	Pastikan bahwa aplikasi / *server web* menyetel header *Content-Type* secara tepat, dan itu menetapkan *header* X-*Content-Type-Options* ke 'nosniff' untuk semua halaman *web*.Jika memungkinkan, pastikan bahwa pengguna akhir menggunakan *browser web standar-compliant.*
6	*Cross -Domain JavaScript Source file inclusion*	2	Refrensi OWASP.
7	*Web browser XSS Protection No. enable*	1938	Refrensi OWASP.
8	*X-Content type Options header missing*	2109	Refrensi OWASP.
Total		6049	

Sumber: Real-time oleh *Open web aplication security project* (OWASP).

4. Kesimpulan

Kesimpulan dari penelitian ini audit, diawali dengan langkah *footprinting*, *scanning vulnerability*, *Reporting analysis*. Kelebihan dari *tool* OWASP dapat melihat *source code* yang di tandai khusus oleh *tool* OWASP. Sistem metodologi yang digunakan *vulnerability assessment* pada penelitian ini pengujian yang dilakukan tanpa mengetahui

struktur rancang bangun pada target yang dituju. *tool* OWASP berhasil menguji kerentanan sistem OJS. Pengujian yang telah dilakukan berhasil mengidentifikasi 3 tingkat kerentanan, yaitu *high*, *medium* dan *low*. Tingkat kerentanan diperoleh dari noktifikasi alert yang ditampilkan oleh *tool* OWASP. Hasil pengujian pada OJS diperoleh *tool* OWASP kerentanan high 70, kerentanan medium 1929 dan 4050 kerentanan *low*. Total celah atau *vulnerability* yang ditemukan berjumlah 6049. Hasil pengujian yang dilakukan menunjukkan bahwa pada OJS versi 2.4.7 memiliki banyak celah atau kerentanan tidak di rekomendasi untuk di gunakan, gunakanlah versi terbaru yang dikeluarkan oleh pihak OJS *Public knowledge project* (PKP).

DAFTAR PUSTAKA

BARGHUTHI, SALEH, ALSUWAIDI, ALHAMMADI, 2017. Information Technology Trends (ITT). *IEEE*. "*Evaluation of Portable Penetration Testing on Smart Cities Applications Using Raspberry Pi* III." (Itt): 25–26. *higher Colleges of Technology Sharjah, United Arab Emirates.*

GHANEM dan BELATON, 2013. *IEEE.Conference, Ieee International, Computer Sciences, and Universiti Sains. "Improving Accuracy of Applications Fingerprinting on Local Networks Using NMAP-AMAP-ETTERCAP as a Hybrid Framework.": 403–7. Sechool of Computer Sciences. Universiti Sains* Malaysia (USM). Penang. Malaysia.

ELIZABETH dan JIMENEZ, 2016. *ITCA-FEPADE*. "*Pentesting on Web Applications Using Ethical Hacking." (503). El Salvador.*

FIRDAUSY, KARTIKA, SAMADRI, dan YUDHANA. A, 2008. TELKOMNIKA "Sistem Informasi Perpustakaan Berbasis Web Dengan *Php* Dan *Mysql*." TELKOMNIKA (*Telecommunication Computing Electronics and Control*) 6(2): 109. Program Studi Teknik Elektro. Universitas Ahmad Dahlan Yogyakarta. Indonesia.

AKHOON, GANAIE, dan KHAZIR, 2018. *IEEE*. *"*Research Data Management in Open Access Journals by Developed Countries." *2018 5th International Symposium on Emerging Trends and Technologies in Libraries and Information Services (ETTLIS): 116–20. Department of Lib and Information Science, University of* Kashmir.

HAN WU, GAO, ZU, 2018. IEEE "An Assessment Approach of the Power System Vulnerability Considering the Uncertainties of Wind Power Integration." *2018 China International Conference on Electricity Distribution (CICED) (201804270000656): 741–45. Student Member IEEE.*

LETIAN, JIANMING, JING dan GOJUN, 2014. *IEEE. "PVDF:* An Automatic Patch-Based Vulnerability Description and Fuzzing Method*". School of Computer, Wuhan University, China, Key Lab of Aerospace Information Security and Trusted Computing, Ministry Education, Wuhan University*, China

KURNIAWAN, RIADI. I dan LUTHFI, 2017. *Internasional Journal of Computer Science and Information Security (IJCSIS). "*Forensic Analysis and Prevent of Cross Site Scripting in Single Victim Attack Using Open Web Application Security Project (OWASP) Framework*." Journal of Theoretical and Applied Information Technology 95(6): 1363–71. Department of Informatics Engineering, Department of Information System, Department of Informatics Engineering. Islamic Univesity of Indonesia, Ahmad Dahlan University* Yogyakarta, Indonnesia.

KURNIAWAN. RIADI. I, 2018. IJCSIS. "Security Level Analysis of Academic Information System Based on Standart ISO27002: 2013 Using SSE-CMM*." (January). Department of Informatics Engineering Islamic Univesity of Indonesia Yogyakarta, Indonesia, Department of Information System. Ahmad Dahlan University Yogyakarta, Indonesia.*

MANALU, WILLY, dan PRIATNA, 2017. *IEEE. Internasional Conference on Eletrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON). "*Development of Review Rating and Reporting in Open Journal System*.": 842–45. School of Computer Science, Bina Nusantara University Jakarta, Indonesia.*

MUTEMWA, MTSWENI, ZIMBA, 2018. "Integrating a Security Operations Centre with an Organization' s Existing Procedures, Policies and Information Technology Systems*." 2018 International Conference on Intelligent and Innovative Computing Applications (ICONIC): 1–6. Department of Peace, Safety and Security The Council of Scientic and Industrial Resarch Pretoria, South Africa.*

NAIK, dan JENKIN, 2018. "Discovering Hackers by Stealth: Predicting Fingerprinting Attacks on Honeypot Systems*." 2018 IEEE International Systems Engineering Symposium (ISSE): 1–8. Defence school of*

Comunications and Information System Ministry of Defence, United Kingdom.

RIADI. I dan UMAR. R, 2016. PPs UMY. "Analisis Forensik Serangan *SQL INJECTION* Menggunakan Metode Statis Forensik.": 102–3. Sistem Informasi, Teknik Informatika, Magister Teknik Infrotmatika. Universitas Ahmad Dahlan Yogyakarta, Indonesia

GORBENKO, ROMANOVSKY, TARASYUK, BILOBORODOY, 2017. *IEEE. "*Experience Report: Study of Vulnerabilities of Enterprise Operating Systems*."School of Computing, Creative Technologies & Engineering, Leeds Backeet University, Leeds, United Kingdom (UK).*

UN, MENG, GAO, BO HU, 2018. *IEEE. "*Universal Framework for Vulnerability Assessment of Power Grid Based on Complex Networks*.": 136–41. School of Information Science and Engineering, Northeastern University, Shenyang. State Grid Huludao Electric Power Supplay Company, Huludao.*

BO WANG, dan XUNTING WANG. 2018. *IEEE. "*Vulnerability Assessment Method for Cyber Physical Power System Considering Node Heterogeneity." *2018 IEEE Innovative Smart Grid Technologies - Asia (ISGT Asia): 1109–13. Department of Electrical Engineering, Wuhan University, Wuhan China.*

YUDHANA. A., RIADI. I., RIDHO. F, 2018. IJACSA *Internasional Journal of Advanced Computer Science and Applications. "*DDoS Classification Using Neural Network and Naïve Bayes Methods for Network Forensics." *9(11): 177–83. Department of Electrical Engineering, Department of Information System, and Department of Informatics Engineering. Universitas Ahmad Dahlan Yogyakarta, Indonesia. Vol 9. No.11.*

YUNANRI, RIADI. I, YUDHANA. A, 2018. JURTI "Analisis Deteksi *Vulnerability* Pada *Webserver Open Journal System* Menggunakan *OWASP Scanner*." Magister Teknik Informatika, Sistem Informasi, Teknik Elektro. Universitas Ahmad Dahlan Yogyakarta, Indonesia. Vol 2, Juni 2018.