PENYUSUNAN RENCANA PENGUJIAN KEAMANAN APLIKASI BERBASIS *WEB*

Falahah

Program Studi Teknik Informatika, Fakultas Teknik, Universitas Widyatama
Jl. Cikutra no.204 A, Bandung. 40124
Telp : +62-22-7275855 ext 228, Fax : +62-22-720299
E-mail : falahah@widyatama.ac.id

Abstrak

Keamanan informasi merupakan isu yang sangat penting. Banyaknya ancaman keamanan pada aplikasi berbasis web membuat pengembang aplikasi berbasis web harus juga memperhatikan aspek keamanan dalam pembangunan dan pengujiannya. Kendala pada adopsi keamanan sistem ini adalah kerap kali aspek keamanan tidak dirancang secara khusus pada tahap awal, tetapi diminta diuji di tahap akhir. Pada beberapa kasus, seringkali pengujian keamanan dilakukan setelah aplikasi digunakan. Penelitian ini bertujuan mengusulkan pendekatan penyusunan rancangan pengujian keamanan sistem dengan mengadopsi framework OWASP sebagai alat bantu untuk menilai risiko keamanan sistem. Pendekatan yang digunakan adalah menerapkan prinsip-prinsip Secure SDLC, yaitu memasukkan aspek-aspek keamanan saat menjalankan SDLC. Hasil penelitian menunjukkan bahwa kita dapat menggunakan kerangka OWASP sebagai alat bantu untuk menentukan prioritas risiko, kemudian menyiapkan rencana pengujian berdasarkan prioritas risiko tersebut. Rencana pengujian ini merupakan dokumen pelengkap dari rancangan pengujian aplikasi secara keseluruhan.

Kata kunci: *security, web application, SSDLC, framework, OWASP, testing plan*

1. PENDAHULUAN

Pada rangkaian siklus hidup pengembangan sistem atau SDLC (*System Development Life Cycle*), fase pengujian adalah fase yang cukup kritikal. Fase ini menjadi penting karena akan menentukan apakah sistem sudah dibangun sesuai dengan kebutuhan dan layak pakai. Seperti halnya fase-fase lain pada SDLC (analisis, perancangan, pembangunan, pengujian, implementasi, dan pemeliharaan), fase pengujian juga dituntut untuk dilakukan dengan terencana dan diakhiri dengan membuat laporan pengujian. Pengujian yang dilakukan pada sistem atau aplikasi juga idealnya harus terencana dan mampu mencakup isu-isu penting pada aplikasi, seperti isu fungsionalitas, kinerja, kehandalan, dan tentunya keamanan sistem. Meskipun fase pengujian ini penting, pada kenyataannya menjalankan pengujian tidaklah mudah. Hal ini dikarenakan terbatasnya sumber daya serta banyaknya hal yang harus diuji. Oleh karena itu, pada praktisnya, pengembang aplikasi biasanya hanya menguji isu-isu penting yang menjadi prioritas aplikasi tersebut.

Di lain pihak, keamanan merupakan aspek penting pada aplikasi, apalagi aplikasi berbasis *web* yang dapat berjalan di internet dan diakses oleh banyak orang. Tingginya ancaman terhadap aplikasi berbasis *web* ini sudah banyak diselidiki oleh berbagai penelitian, lembaga riset, ataupun komunitas independen. Salah satunya adalah OWASP (*Open Web Application Security Project*) yang merupakan komunitas daring yang mengkhususkan diri pada bidang keamanan aplikasi berbasis *web*. Setiap tahun OWASP merilis daftar ancaman keamanan yang dapat dijadikan referensi bagi pengembang aplikasi untuk diperhatikan dan diantisipasi oleh aplikasi yang akan dibuat [4]. Selain merilis daftar risiko keamanan, OWASP juga menyertakan beberapa alat bantu untuk memudahkan melakukan analisis risiko keamanan. Berdasarkan kerangka kerja OWASP ini, kita dapat menentukan prioritas risiko yang harus diantisipasi dan menyiapkan mekanisme pengujian untuk risiko tersebut.

Prinsip-prinsip pada ISO/IEC 27001 mensyaratkan perlunya diterapkan manajemen risiko dalam mengidentifikasi kelemahan sistem dan mengantisipasinya dengan sekumpulan tindakan [1]. Adopsi prinsip-prinsip keamanan informasi pada siklus hidup pembangunan sistem, yang kemudian dikenal sebagai *Secure* SDLC (SSDLC), menghasilkan sekumpulan peraturan / *rule* yang harus diperhatikan pada pembangunan sistem. Peraturan atau *rule* tersebut terdiri atas 21 aspek [2].

Pada pendekatan SSDLC, identifikasi kebutuhan sistem tidak hanya mencakup kebutuhan fungsionalitas dan non fungsionalitas, tetapi juga kebutuhan keamanan. Seperti halnya kebutuhan pada software, pada SSDLC, identifikasi kebutuhan keamanan dapat dikategorikan menjadi *functional security requirement, non-functional security requirement, derived security requirement, user stories, dan abuse case*[3].

Terkait dengan pendekatan SSDLC di atas, pada saat proses pengujian, selain dilakukan pengujian fungsional, juga perlu dilakukan pengujian keamanan sistem. Dasar pengujian keamanan adalah rancangan keamanan sistem yang sudah didefinisikan berdasarkan identifikasi kebutuhan kemanan (*security requirement*). Dikarenakan proses pengujian itu sendiri tidak mudah dan cukup menyita sumber daya, maka untuk mengoptimalkan ketersediaan sumber daya perlu dilakukan prioritas dan perencanaan pengujian. Perencanaan pengujian dibuat mengacu pada hasil perancangan keamanan sistem. Sedangkan, perancangan keamanan sistem mengacu pada risiko kelemahan sistem (*system vulnerability).* Oleh karena itu, perancangan keamanan yang sistem yang baik perlu dimulai dengan kajian risiko keamanan sistem.

Idealnya, aplikasi *web* yang dibuat seharusnya dapat mengatasi semua risiko seperti 10 daftar risiko OWASP di atas. Tetapi, proses yang diperlukan untuk penyiapan sistem agar aman, serta proses pengujiannya, akan sangat rumit. Proses pengujian yang dilakukan untuk *Injection* dan *Broken access control*, misalnya, tidak sama. Satu jenis risiko mungkin memerlukan beberapa jenis pengujian. Oleh karena itu, agar dapat diterapkan secara efektif, diperlukan *assessment* untuk melihat risiko mana yang paling berpeluang untuk terjadi. Hal ini karena risiko dapat dipengaruhi oleh kondisi organisasi/lingkungan, dan karakteristik aplikasi itu sendiri. Guna mengatasi hal tersebut, OWASP juga menyediakan metode untuk menilai risiko (*risk assessment*).

2. KAJIAN PUSTAKA

Pada proses SSDLC, manajemen keamanan sistem informasi terintegrasi penuh dengan proses SDLC itu sendiri. Tindakan pengamanan sistem meliputi satu proses utuh mulai dari identifikasi kebutuhan keamanan (*security requirement*), perancangan keamanan, pembangunan fitur-fitur keamanan sistem, dan pengujian kemanan. Pada proses pengujian keamanan, diperlukan perencanaan pengujian dengan mengacu pada perancangan keamanan. Pada perancangan keamanan, diperlukan spesifikasi khusus yang harus dipenuhi sesuai dengan kebutuhan keamanan. Kebutuhan keamanan mengacu pada hasil kajian risiko keamanan. Oleh karena itu, tahap awal pada identifikasi kebutuhan keamanan adalah kajian risiko keamanan. Kajian risiko keamanan ini dapat dilakukan dengan menggunakan pendekatan atau kerangka kerja tertentu. Salah satunya menggunakan pendekatan OWASP khusus untuk aplikasi berbasis *web*.

Assessment risiko berdasarkan *framework* OWASP dilakukan melalui sistem skoring yang disusun berdasarkan *threat agent factor* (actor/pihak yang berpotensi mengancam), *vulnerability factor* (titik kelemahan sistem), dan *impact* (potensi kerugian/dampak dari serangan tersebut). *Threat agent factor* mengukur seberapa besar peluang terjadinya ancaman risiko berdasarkan kapasitas kemampuan agent (pihak) yang menyerang sistem. *Threat agent* dibagi menjadi *Skill level*, *motive*, *opportunity* dan *size*. Masing-masing *threat agent* pada masing-masing parameter, diberi skor. *Vulnerability factor* mengukur seberapa faktor kelemahan tesebut mudah ditemukan, yang dapat diidentifikasi melalui beberapa komponen yaitu : *Ease of discovery* (seberapa sulit kelemahan tersebut dapat ditemukan), *Ease of exploit* (seberapa mudah *thread agent* mengeksploitasi kelemahan ini), *awareness* (seberapa tingkat kepedulian pengelola sistem terhadap jenis serangan tersebut), dan *intrusion detection* (sejauh mana usaha mendeteksi adanya serangan tersebut diterapkan pada sistem) [5].

Tabel 1 menampilkan skor untuk masing-masing parameter[5]. Sebagai contoh, untuk parameter *skill level*, kondisi jika aplikasi berjalan pada lingkungan yang memiliki keahlian penetrasi keamanan tinggi akan mendapat skor 9.

Tabel 1. Parameter *Threat Agent Factor* dan *Vulnerability Factor*

Threat Agent Factor

Parameter	***Condition***	***Score***
Skill level	*Security penetration skill*	*9*
	Network and programming skill	*6*
	Advanced computer user	*5*
	Technical skills (average)	*3*
	No technical skill	*1*
Motive	*Low / no reward*	*1*
	Possible reward	*4*
	High reward	*9*
Opportunity	*Full access/expensive*	*0*
	Special access/ resource required	*4*
	Some access or resource required	*7*
	No access or resource required	*9*
Size	*Developers*	*2*
	System administrator	*2*
	Intranet users	*4*
	Partner	*5*
	Authenticated user	*6*
	Anonymous internet user	*9*

Vulnerability Factor

Parameter	***Condition***	***Score***
Ease of discovery	*Practically impossible*	*1*
	Difficult	*3*

Parameter	*Condition*	*Score*
	Easy	*7*
	Automated tools available	*9*
	Theoretical	*1*
	Difficult	*3*
	Easy	*5*
	Automated tools available	*9*
Awareness	*Unknown*	*1*
	Hidden	*4*
	Obvious	*6*

Parameter	*Condition*	*Score*
	Public knowledge	*9*
Intrusion	*Active detection in application*	*1*
detection	*Logged and reviewed*	*3*
	Logged without review	*8*
	Not logged	*9*

Impact mengukur sejauh mana dampak akibat serangan/gangguan terhadap sistem. *Impact* diukur berdasarkan dua aspek yaitu aspek teknis (terganggunya layanan secara teknis) dan aspek bisnis (pengaruh terhadap kondisi finansial organisasi). Pada kerangka kerja OWASP, *impact* dibagi menjadi *technical impact* dan *business impact.*

Technical Impact mengukur dampak secara teknis akibat adanya serangan/gangguan terhadap sistem. *Technical impact* dibagi menjadi beberapa parameter yaitu *loss of confidentiality, loss of integrity, loss of availability*, dan *loss of accountability*. *Business impact* mengukur pengaruh serangan atau kegagalan terhadap bisnis organisasi secara keseluruhan. *Business impact* diukur melalui 4 parameter yaitu *financial damage, reputation damage, non-compliance,* dan *privacy violation.* Tabel 2 menampilkan masing-masing parameter untuk *Technical Impact* dan *Business Impact* [5].

Tabel 2. *Technical Impact* dan *Business Impact*

Technical Impact

Parameter	**Condition**	**Score**
Loss of confidentiality	*Minimal/non sensitive data disclosed*	2
	Minimal critical data disclosed	6
	Extensive non sensitive data disclosed	6
	Extemsove critical data disclosed	7
	All data disclosed	9
Loss of integrity	*Minimal slighty corrupt data*	1
	Minimal seriously corrupt data	3
	Extensive slightly corrupt data	5
	Extensive seriously corrupt data	7
	All data totally corrupt	9
Loss of availability	*Minimum secondary services interrupted*	1
	Minimum primary services interrupted	5
	Extensive secondary services interrupted	5
	Extensive primary services interrupted	7
	All service completely lost	9
Lost of accountability	*Fully traceable*	1
	Possibly traceable	7
	Completely anonymous	9

Business Impact

Parameter	**Condition**	**Score**
Financial damage	*Less than the cost to fix vulnerability*	1
	Minor effect on annual profit	3
	Significant effect on annual profit	7
	Bankruptcy	9
Reputation damage	*Minimal damage*	1
	Lost of major account	4
	Lost of goodwill	5
	Brand damage	9
Non compliance	*Minor violation*	2
	Clear violation	5
	High profile violation	7
Privacy violation	*One individual*	3
	Hundreds of people	5
	Thousands of people	7
	Millions of people	9

Assessment dilakukan dengan cara memberikan skor untuk masing-masing ancaman kelemahan, berdasarkan kondisi lingkungan. Misalnya, untuk satu jenis risiko "pencurian *database* dari *data center*", dilakukan penilaian sebagai berikut:

1) *Likelihood* (peluang terjadi), dihitung dari 2 hal yaitu:
    a. *Threat agent factor*, masing-masing diberi skor untuk 4 paramenter, misalnya untuk *skill level* : 4 (hanya dapat dilakukan oleh *advanced user*), *motive*: 1(*low or no reward*, artinya hasil pencurian tidak bernilai tinggi),
    b. *Vulnerability factor*, diberi skor untuk 4 parameter
2) *Overall likelihood* dihitung dari rata-rata seluruh skor untuk 8 parameter (4 untuk *threat agent factor* dan 4 untuk *vulnerability factor*). *Impact* (dampak), dihitung dari 2 hal yaitu:
    a. *Technical impact*, yang terdiri atas 4 parameter
    b. *Business impact*, terdiri atas 4 parameter

3) *Overall impact* dihitung dari rata-rata seluruh skor. Hasil perhitungan untuk *likelihood* dan *Impact* dapat dilihat pada Gambar 1.

Selanjutnya, tingkat risiko dipetakan sesuai dengan nilai risiko yang diperoleh dari model perhitungan di atas, menjadi tiga kategori yaitu *Low*, *Medium* dan *High*. Jika hasil perkalian *impact* dan *likelihood* kurang dari 3, maka dikategorikan *Low*, Jika lebih besar dari 6 dikategorikan *High*, sedangkan nilai antara 3 dan 6 dikategorikan *Medium*.

Pada contoh seperti pada Gambar 1, nilai risiko untuk pencurian *database* dari *data center* adalah 3.375 x 2.250 = 7.59, sehingga termasuk pada kategori tingkat risiko tinggi (antara 6 hingga 9).

Risk: Full database theft from datacenter

Likelihood								
Threat agent factors					Vulnerability factors			
Skill level	Motive	Opportunity	Size		Ease of discovery	Ease of exploit	Awareness	Intrusion detection
4 - Advanced computer user	1 - Low or no reward	access or resources required	5 - Partners		3 - Difficult	3 - Difficult	4 - Hidden	3 - Logged and reviewed
Overall likelihood:				3.375	MEDIUM			

Technical Impact					Business Impact			
Loss of confidentiality	Loss of integrity	Loss of availability	Loss of accountability		Financial damage	Reputation damage	Non-compliance	Privacy violation
2 - Minimal non-sensitive data disclosed	0 -	0 -	9 - Completely anonymous		1 - Less than the cost to fix the vulnerability	1 - Minimal damage	0 -	5 - Hundreds of people
Overall technical impact:		2.750	LOW		Overall business impact:		1.750	LOW
Overall impact:				2.250	LOW			

Gambar 1. Contoh Hasil Assessment Risiko

Langkah *assessment* di atas kemudian dilakukan untuk setiap potensi risiko. Jika mengacu pada daftar 10 risiko berdasarkan rilis OWASP, maka perlu dilakukan *assessment* untuk 10 risiko. Tentunya dapat diputuskan untuk mengurangi daftar risiko, misalnya hanya 5 atau 6 risiko dari 10 daftar risiko sesuai rilis OWASP tersebut. Setelah dilakukan *assessment* pada setiap potensi risiko, kemudian disusun prioritas risiko berdasarkan skor, dan ditentukan risiko dengan prioritas tinggi. Hasilnya seperti contoh pada Tabel 3.

Tabel 3. Contoh Prioritas RISiko

No	*Risk Type*	*Risk Score*	*Category*
1	*Injection*	7.8	*High*
2	*Sensitive Data exposure*	4	*Medium*
3	…..	…	….
4			

Penelitian-penelitian terkait penyusunan pengujian keamanan sejauh ini lebih berfokus pada isu dan tantangan pengujian keamanan [6], jenis-jenis serangan keamanan dan model ancaman [7], strategi untuk keamanan aplikasi *web* [8] dan assessment keamanan aplikasi *web* menggunakan teknik penetrasi. Dari 4 penelitian tersebut, hasil penelitian yang memadai adalah yang terkait *assessment* keamanan aplikasi. Tetapi, penelitian ini tidak membahas bagaimana membuat rencana pengujian keamanan aplikasi.

3. HASIL DAN PEMBAHASAN

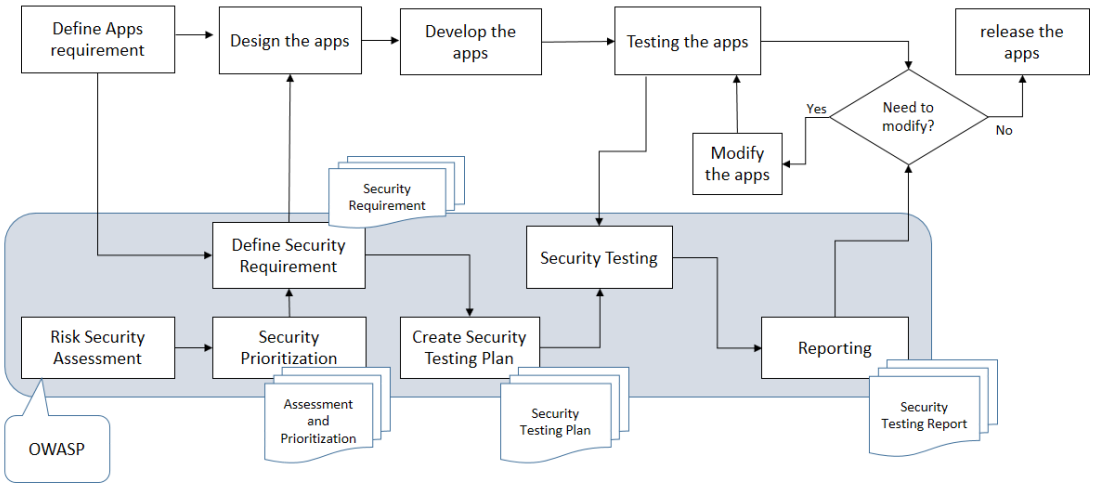
Berdasarkan prinsip-prinsip dari SSDLC dan kerangka kerja OWASP, kemudian diusulkan kerangka kerja pengujian keamanan untuk aplikasi berbasis *web*. Prinsip penyusunan kerangka rencana pengujian ini mengacu pada konsep tata kelola dari COBIT 5, yaitu meliputi produk dan proses. Hal ini karena adanya rencana kegiatan merupakan salah satu unsur pada tata kelola. Proses mengacu pada aktivitas yang dilakukan dalam penyusunan rencana, dan produk mengacu pada luaran pekerjaan penyusunan rencana.

Yi mendefinisikan aktivitas-akitivitas yang dilakukan pada pengujian[9]. Beberapa aktivitas tersebut akan menghasilkan dokumen sebagai produk kegiatan tersebut. Tabel 2 menampilkan aktivitas dan produk yang dihasilkan pada SSDLC.

Tabel 2. Aktivitas Pengujian dan Produk yang dihasilkan

NO	Aktivitas	Keterangan	Cara/metode/pendekatan	Produk/Output
1	*Risk Security Assessment*	melihat risiko keamanan apa saja yang mungkin terjadi dan berapa besar peluang terjadinya.	Menggunakan *framework security assessment*, misalnya OWASP	Hasil *assessment* risiko keamanan dan prioritas keamanan
2	*Security Prioritization*,	berdasarkan hasil *assessment* kemudian dibuat prioritas isu keamanan yang akan dicakup aplikasi	Menentukan prioritas berdasarkan batas atas skor risiko (dari hasil langkah 1)	
3	*Define security requirement*	mendefinisikan kebutuhan keamanan yang harus disertakan pada saat membangun sistem	Menjelaskan spesifikasi sistem yang harus tersedia agar kebutuhan keamanan dapat dipenuhi, misalnya data harus dienkripsi, *password* harus diganti setiap 6 bulan, dan seterusnya.	Kebutuhan keamanan aplikasi (*Security requirement*)
4	*Create Security Plan*	membuat rencana uji untuk setiap jenis risiko keamanan yang sudah disesuaikan dengan prioritas.	Menyusun langkah-langkah pengujian, data uji, dan jadwal uji, teknik uji.	Rencana Pengujian Keamanan (*Security testing plan*)
5	*Security Testing*	melakukan uji keamanan aplikasi sesuai dengan rencana uji keamanan	Menjalankan pengujian sesuai dengan rencana (langkah 4), dan membuat laporan.	*Security Testing Report* (hasil pengujian keamanan)

Setelah hasil uji keamanan dilaporkan, akan terdapat kemungkinan aplikasi harus dimodifikasi. Jika aplikasi dimodifikasi, maka proses uji harus diulang kembali, demikian seterusnya hingga aplikasi siap rilis. Secara konseptual, proses pengujian keamanan aplikasi dan dokumen-dokumen yang dihasilkan, serta keterkaitan siklus pengujian keamanan sistem berbasis *web* dengan siklus SDLC, dapat dilihat pada Gambar 3. Daerah yang diarsir menggambarkan ruang lingkup kegiatan pengujian. Penerapan OWASP dapat dilakukan pada aktivitas pertama yaitu *Risk Security Assessment*.



Gambar 3. Kerangka Pengujian Keamanan Aplikasi

Berdasarkan hasil analisis risiko keamanan, kemudian dibuat rancangan pengujian keamanan untuk masing-masing risiko. Jika kemudian ditemukan 3 jenis ancaman keamanan yang memiliki risiko tinggi, maka pengembang harus menyiapkan 3 jenis rancangan pengujian keamanan. Pada contoh ini, misalnya akan diambil contoh rancangan pengujian keamanan untuk risiko *SQL Injection*. Rencana pengujian memuat hal-hal seperti:

a. Jenis Risiko : ***SQL Injection***
b. Tujuan Pengujian : Memeriksa adanya peluang SQL Injection pada aplikasi

c. Teknik Deteksi:
   1) Identifikasi titik akses aplikasi ke *database*, misalnya otentifikasi *user* (*login*), pencarian data (melalui *search engine*) atau daftar produk pada situs *e-commerce*.
   2) Identifikasi semua *field* masukan yang umumnya dikirimkan ke SQL Query, termasuk *field* yang tersembunyi menggunakan *method* POST
   3) Lakukan pengujian untuk semua *field*, dengan mengidentifikasi adanya peluang:
      - Parameter yang dapat disisipkan (*injectable*)
      - *Database finger-printing* (mengidentifikasi tipe dan versi *database*)
      - Mengetahui skema *database*

d. Jadwal Pengujian, seperti contoh 1 (tabel 3.a)
e. Tim Penguji: mencatat nama-nama, peranan dan kontak para petugas yang melakukan pengujian, seperti contoh 2 (tabel 3.b)
f. Teknik Pengujian Secara Otomatis: menjelaskan pengujian yang dilakukan secara otomatis melalui alat bantu atau *tools*, agar dijelaskan alat bantu yang digunakan, seperti pada contoh 3.
g. Teknik Pengujian Manual, menjelaskan aktivitas pengujian yang dilakukan secara manual, seperti pada contoh 4

Tabel 3a, 3b, dan 3c merupakan contoh dari hasil pelaksanaan pengujian sesuai rencana di atas.

Tabel 3a. [Contoh 1] Jadwal Pengujian

Jenis Pengujian : Injectable Parameter
Petugas Pelaksana Pengujian : NAMA

Parameter uji	**SQL Uji**	**Tgl Uji**	**Hasil *)**
Username/ password	Dituliskan *script* untuk pengujian	Dd/mm/yyyy	Berhasil/gagal
…..	…..	….	…

*) Berhasil : sql injection dapat dilakukan, berarti sistem harus diamankan; Gagal : sql injection tidak dapat dilakukan, berarti sistem aman terhadap serangan sql injection.

Tabel 3b. [Contoh 2] TIM PENGUJI

NRK/Nama	***Role***	***Email-address***	**Telpon**
XXXX	*Admin*	aa@bb.com	XXXX

Tabel 3c. [Contoh 3] Rencana Pengujian Secara Otomatis

Nama *Tools*	**Vendor/organisasi/versi**	**Tujuan Penggunaan Tools**
SQLNinja THC Hydra	http://sqlninja.sourceforge.net http://www.thc.org/thc-hydra/	*Sql injection* *Bruteforce password*
….	….	…..

Tabel 3.d. [Contoh 4] Rencana Pengujian Secara Manual

No	**Nama Pengujian**	**Deskripsi**
T001	SQL Injection	Penguji akan menjalankan beberapa serangan SQL *Injection* menggunakan akun palsu dan *statement* 0 *OR* ‘1’=’1’.
T002	Captcha	Penguji akan menggunakan fungsi *captch* pada *form login* secara manual.
….	…..	……

4. SIMPULAN DAN SARAN

Berdasarkan hasil diskusi pada bagian sebelumnya dapat disimpulkan bahwa untuk menerapkan keamanan pada pembangunan sistem maka kita harus menerapkan konsep SSDLC yang mensyaratkan identifikasi kebutuhan keamanan dari awal tahapan SDLC. Selanjutnya, dengan bantuan kerangka kerja OWASP kita dapat mengidentifikasi peluang keamanan dan menjadikan masukan bagi penentuan prioritas keamanan. Penyusunan rencana pengujian keamanan didasari oleh hasil penentuan prioritas sehingga kita tidak menghabiskan sumber daya untuk menguji risiko keamanan yang rendah peluangnya. Hasil penelitian ini berupa kerangka kerja penyusunan rencana pengujian keamanan informasi yang terdiri atas dua bagian yaitu produk dan proses. Produknya adalah template dokumentasi *assessment*, perencanaan dan pelaporan hasil pengujian keamanan. Prosesnya terdiri atas 6 proses utama yaitu *assessment, prioritization, requirement, plan, test*, dan *reporting*.

5. DAFTAR RUJUKAN

[1] Utomo M., Ali A.H.N, Affandi I, Pembuatan Tata Kelola Keamanan Informasi Kontrol Akses Berbasis ISO/IEC 27001:2005 Pada Kantor Pelayanan Perbendaharaan Surabaya I, Jurnal Teknik ITS Vol.1, No.1, (Sept 2012)

[2] Banerjee C., Pandey S. K., Software Security Rules: SDLC Perspective, International Journal of Computer Science and Information Security (IJCSIS), Vol. 6, No.1,

[3] Priya S.S., Malarchevi P.D.S.K, Security Deliberation in Software Development Lifecycle, International Conference on Information and Communication Technologies(ICICT-2014)

[4] Rafique S., Humayun M, Gul Z, Abbas A, Javed H, Systematic Review of Web Application Security Vulnerabilities Detection Method, Journal of Computer and Communications ,2015, 3, 28- 40

[5] OWASP Risk Rating Methodology, diakses dari: https://www.owasp.org/index.php/ OWASP_Risk_Rating_ Methodology

[6] Cobit 5 Enabling Process.ISACA.

[7] Li, Y. Conceptual Framework for Security Testing, Security Risk Analysis, and their combination. STV 2012

[8] Izzagire, M., Deception Strategis for web application security: application layer approach and testing platform, Master of Science Information security.

Halaman ini sengaja dikosongkan.