Analisis Dan Perancangan Keamanan Aplikasi Manajemen Layanan Sistem Pemerintahan Berbasis Elektronik

Widi Linggih Jaelani[1], Ira Monellia[2], Muhamad Malik Mutoffar[3]
Departemen Teknik Informatika, Bidang Perencanaan TIK
Universitas Teknologi Bandung dan Dinas Komunikasi dan Informatika
Bandung, Indonesia
e-mail: [1]jaelaniwidi@gmail.com, [2]iramonellia@gmail.com, [3]malik@utb-univ.ac.id

Correspondence : *e-mail*: widilinggih@sttbandung.ac.id

Diajukan: 07 Agustus 2024; Direvisi: 23 Agustus 2024; Diterima: 24 Agustus 2024

Abstrak

Perkembangan Teknologi Informasi dan Komunikasi (TIK) mendorong pemerintah menerapkan Sistem Pemerintahan Berbasis Elektronik (SPBE) yang bertujuan memberikan pelayanan yang efektif dan efisien kepada masyarakat local, aplikasi pengelolaan layanan SPBE menghadapi berbagai tantangan keamanan siber seperti serangan siber, pencurian data, dan pelanggaran hak istimewa, yang dapat menghambat proses pelayanan publik serta membahayakan privasi dan keamanan data pengguna yang ada. Tujuan penelitian ini adalah menganalisis dan merancang keamanan aplikasi manajemen layanan SPBE. Analisis mencakup identifikasi risiko keamanan, evaluasi kontrol keamanan yang ada, dan pengembangan strategi remediasi yang tepat. Desainnya mencakup penerapan solusi keamanan terintegrasi seperti manajemen identitas, kontrol akses, enkripsi data, serta deteksi dan respons insiden keamanan. Pengumpulan data dilakukan melalui penelitian literatur, wawancara dengan pemangku kepentingan di lingkungan Pemerintah Kota Bandung, dan observasi lapangan. Hasil penelitian membuktikan bahwa aplikasi manajemen pelayanan SPBE meningkatkan keamanan dan ketahanan, mendukung penyelenggaraan pemerintahan yang lebih efektif, efisien, dan andal. Pentingnya penerapan strategi keamanan komprehensif untuk mengatasi tantangan keamanan siber di SPBE.

Kata kunci: *SPBE, Keamanan Siber, Manajemen Layanan, Enkripsi Data, Mitigasi Risiko.*

Abstract

The development of Information and Communication Technology (ICT) encourages the government to implement an Electronic-Based Government System (EBGS) which aims to provide effective and efficient services to local communities, EBGS service management applications face various cybersecurity challenges such as cyber attacks, data theft, and privilege violations, which can hinder the public service process and endanger the privacy and security of existing user data. The purpose of this study is to analyze and design the security of EBGS service management applications. The analysis includes identifying security risks, evaluating existing security controls, and developing appropriate remediation strategies. The design includes implementing integrated security solutions such as identity management, access control, data encryption, and security incident detection and response. Data collection was carried out through literature research, interviews with stakeholders in the Bandung City Government environment, and field observations. The results of the study prove that the EBGS service management application improves security and resilience, supports more effective, efficient, and reliable governance. The importance of implementing a comprehensive security strategy to overcome cybersecurity challenges in EBGS.

Keywords*: EBGS, Cyber Security, Service Management, Data Encryption, Risk Mitigation.*

1. Pendahuluan

Perkembangan teknologi informasi dan komunikasi (TIK) telah mendorong pemerintah untuk mengimplementasikan Sistem Pemerintahan Berbasis Elektronik (SPBE) [1]. SPBE merupakan penyelenggaraan pemerintahan yang memanfaatkan TIK untuk memberikan layanan yang efektif dan efisien kepada masyarakat [2]. Salah satu komponen penting dalam SPBE adalah aplikasi manajemen layanan, yang digunakan untuk mengelola proses layanan publik secara terpadu dan terstruktur [3].

Penggunaan aplikasi manajemen layanan SPBE tidak terlepas dari tantangan keamanan siber. Ancaman seperti serangan siber, pencurian data, dan tembus otoritas dapat menghambat proses layanan publik serta membahayakan privasi dan keamanan data pengguna [4], [5]. Oleh karena itu, diperlukan analisis dan perancangan keamanan yang komprehensif untuk menjaga keamanan aplikasi manajemen layanan SPBE.

Penelitian ini bertujuan untuk menganalisis dan merancang keamanan aplikasi manajemen layanan SPBE. Analisis akan mencakup identifikasi risiko keamanan, evaluasi kontrol keamanan yang ada, serta pengembangan strategi mitigasi yang sesuai. Perancangan akan melibatkan implementasi solusi keamanan yang terintegrasi, termasuk manajemen identitas, kontrol akses, enkripsi data, dan deteksi serta respons terhadap insiden keamanan [6], [7]. Hasil penelitian ini diharapkan dapat meningkatkan keamanan dan ketahanan aplikasi manajemen layanan SPBE, sehingga dapat mendukung penyelenggaraan pemerintahan yang efektif, efisien, dan terpercaya.

2. Metode Penelitian

Penelitian ini menggunakan pendekatan kualitatif untuk menganalisis dan merancang keamanan aplikasi manajemen layanan SPBE. Metode kualitatif dipilih karena dapat memberikan pemahaman yang lebih mendalam dan komprehensif mengenai isu-isu keamanan siber [8].

Pengumpulan data dalam penelitian ini dilakukan melalui studi literatur, wawancara dengan pemangku kepentingan dilingkungan pemerintahan Kota Bandung, dan observasi lapangan. Studi literatur dilakukan untuk mengidentifikasi *best practices*, standar, dan regulasi terkait keamanan aplikasi SPBE. Wawancara dilakukan dengan pihak-pihak yang terlibat dalam pengembangan dan pengelolaan aplikasi manajemen layanan SPBE dilingkungan Pemerintah Kota Bandung, seperti pengembang, administrator, dan pengguna. Observasi lapangan dilakukan untuk memahami konteks dan proses bisnis aplikasi manajemen layanan SPBE Kota Bandung.

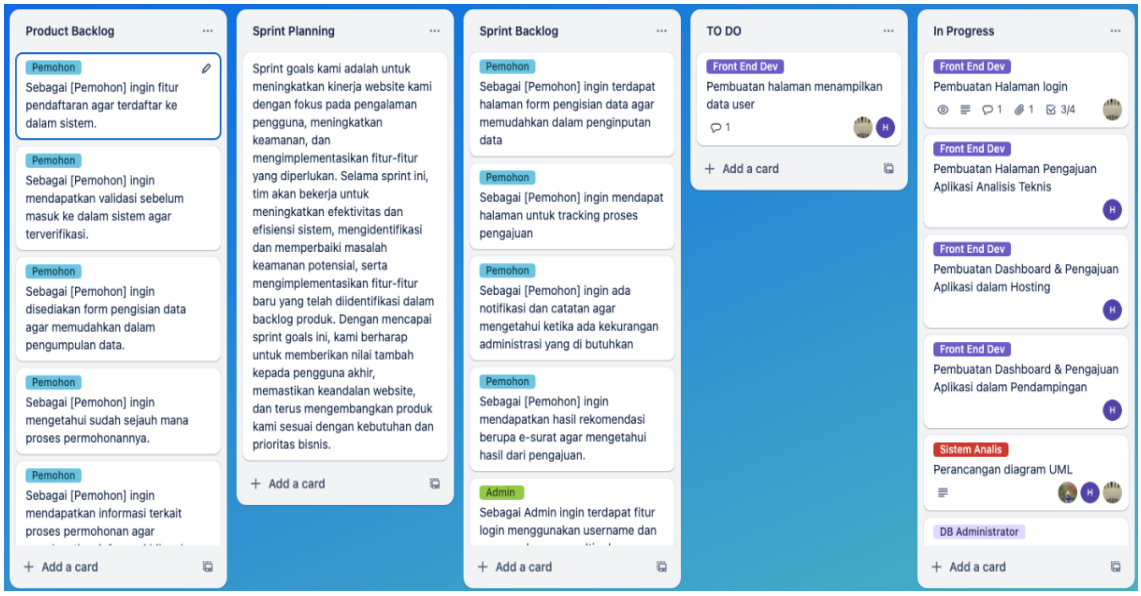
Pengembangan perangkat lunak menggunakan metode *scrum*. *Scrum Guide* memisahkan setiap acara dengan komponen bahasan masing-masing serta memberikan durasi tertentu. Kurangnya sumber daya yang terlibat serta pembagian waktu, yang dirasa akan lebih ringkas apabila ada beberapa acara yang digabungkan, membuat praktik acara pada *scrum* sedikit dimodifikasi oleh tim [9]

3. Hasil dan Pembahasan

Bagian ini membahas kerangka kerja keamanan yang diimplementasikan didalam model *scrum*. Pada tahapan ini data hasil penelitian akan dimunculkan berdasarkan angket dari tanggapan responden, yang bertujuan untuk meningkatkan metodologi dengan memetakan prinsip-prinsip dalam manajemen resiko, yang bisa meningkatkan keberhasilan proyek yang hasilnya ini belum diverifikasi dengan mengujinya didalam kehidupan nyata. Tujuan utama penelitian ini dari adalah agar bisa meningkatkan mekanisme manajemen risiko di *scrum* dan untuk meningkatkan tingkat keberhasilan proyek *Scrum*. [10]

3.1. Scrum Board

Sesuai dengan IEEE Std 829.1-2021 tentang Penerapan Kerangka Kerja *Scrum*, *Scrum Board* digunakan untuk memvisualisasikan alur kerja tim *Scrum*, memfasilitasi pembagian pekerjaan, dan membantu pelacakan kemajuan proyek [11]. IEEE Std 16326-2019 tentang Proses Manajemen Proyek Perangkat Lunak juga menyoroti penggunaan *Scrum Board* sebagai salah satu praktik manajemen proyek dalam konteks *Agile* [12]. IEEE Std 2675-2017 tentang Praktik Rekomendasi untuk Adopsi Praktik Rekayasa dan Manajemen *Agile* menekankan peran *Scrum Board* dalam meningkatkan visibilitas dan kolaborasi tim dalam pengembangan perangkat lunak yang bersifat iteratif dan inkremental [13].



Gambar 1. Papan Kerja *Scrum* Menggunakan Aplikasi *Trello*

3.2. Analisis Kerentanan

Analisis kerentanan merupakan proses identifikasi, evaluasi, dan mitigasi kelemahan atau celah keamanan dalam sistem, perangkat lunak, atau organisasi. Sesuai dengan IEEE Std 1074-2006, Analisis Kerentanan merupakan salah satu aktivitas penting dalam proses pengembangan perangkat lunak yang bertujuan untuk mengurangi risiko dan meningkatkan keamanan sistem [14]. Berikut merupakan hasil sampel analisis kerentanan dari 7 aplikasi yang ada di lingkungan Pemerintah Kota Bandung.

Tabel 1. Analisis Kerentanan Aplikasi

		Aplikasi	A	B	C	D	E	F	G
No	**Insiden**	**Jumlah Insiden**	**Nilai Insiden**	**Nilai Insiden**	**Nilai Insiden**	**Nilai Insiden**	**Nilai Insiden**	**Nilai Insiden**	**Nilai Insiden**
1	SQL Injection	2	0	0	1	0	0	0	1
2	Insecure Direct Object Reference (IDOR)	2	0	0	1	0	0	1	0
3	XSS	4	0	0	1	0	1	1	1
4	HTML Injection	3	0	0	1	0	0	1	1
5	No Rate Limit	7	1	1	1	1	1	1	1
6	Weak Password Policy	5	1	1	1	0	0	1	1
7	Clickjacking	5	1	0	1	1	1	0	1
8	Improper Error Handling	3	1	1	1	0	0	0	0
9	Insecure Data Storage	1	0	0	1	0	0	0	0
10	No Bussiness Logic Data Validation	1	0	0	1	0	0	0	0
11	Ddos	0	0	0	0	0	0	0	0
12	Unristricted *File Upload*	2	0	0	0	0	1	1	0
13	Vulnerable And Component Outdated	2	0	1	0	0	0	0	1
14	SSTI (Server Side Template Injection)	0	0	0	0	0	0	0	0
15	Sensitive Data Exposure	1	0	1	0	0	0	0	0

16	Security Misconfig	0	0	0	0	0	0	0	0
17	Serve Side *Request* Forgery	0	0	0	0	0	0	0	0
18	Cryptographic Failures	0	0	0	0	0	0	0	0
19	CSRF	0	0	0	0	0	0	0	0
20	Excessive Data Exposure	0	0	0	0	0	0	0	0
21	Host Header Injection	1	1	0	0	0	0	0	0
22	Sensitive *File* Disclosure	2	1	0	0	0	0	0	1
23	Directory Traversal	1	0	1	0	0	0	0	0

3.3. Threat Modelling

Threat Modeling merupakan proses untuk mengidentifikasi, menganalisis, dan mengevaluasi ancaman potensial yang dapat berdampak pada sistem, aplikasi, atau organisasi. Sesuai dengan IEEE Std 27034-1-2019, *Threat Modeling* merupakan komponen penting dalam proses manajemen keamanan perangkat lunak, yang bertujuan untuk memahami dan mengurangi risiko keamanan [15]. Berdasarkan hasil analisis kerentanan yang ada maka didapatkan data sebagai berikut :

Tabel 2. Pemodelan Ancaman

Solusi	Insiden	Solusi	Insiden
Pastikan beberapa field data yang ada di *database* menggunakan *cryptography*		Validasi inputan pengguna, terkhusus *special character* (‘”;). Juga dapat melakukan encode pada inputan pengguna. Dapat juga dengan melakukan pembatasan inputan pengguna (jika hal tersebut memungkinkan) Jangan gunakan *query* mentah pada coding sistem	*SQL Injection*
Pastikan untuk *access control* gunakan salah satu metode ini (*dictionary, rolebase, mandatory*) sesuaikan dengan permasalahan yang ada di sistem		Terapkan kontrol akses pada masing-masing pengguna. Pastikan hanya pengguna yang berwenang yang dapat melihat data atau menjalankan fungsi yang dimaksudkan pada sebuah fitur	*IDOR / Broken Access Control*
Implementasikan Captcha dengan benar di setiap *login*	*No Rate Limit*	Terapkan dan tetapkan semua pesan keluaran termasuk pesan kesalahan pada sistem. Agar ketika terjadi kesalahan pada sistem, sistem tidak akan menampilkan pesan kesalahan yang memberikan informasi mengenai sistem aplikasi	*Improper Error Handling*
Menentukan rate limit pada *backend* sistem (Membatasi jumlah *request*)		Menghapus dependensi, fitur, komponen dan dokumentasi yang tidak digunakan oleh sistem aplikasi. Juga lakukan *update* library yang digunakan menjadi versi yang terbaru dan pastikan tidak ada security issues	*Vulnerable and Outdated Component*
Menambahkan header setiap *request* : - *X-FRAME-OPTIONS : DENY* - *X-FRAME-OPTIONS : SAMEORIGIN* - *X-FRAME-OPTIONS : ALLOW* FROM *url* - *Content-Security-Policy : Frame-ancestors 'none'* - *Content-Security-Policy : Frame-ancestors 'self'* - *Content-Security-Policy : Frame-ancestors 'url'*	*Clickjacking*	Validasi inputan pengguna, pastikan inputan pengguna divalidasi atau dibersihkan sebelum di proses. Tolak inputan mencurigakan seperti spesial karakter pada directory traversal attack ini ('../','..\'). Jika memungkinkan, hindari penggunaan input pengguna untuk mengakses *file*. Jika harus, pastikan inputan diperiksa dengan kuat dan tidak berisi urutan traversal direktori.	*Directory Traversal*

Menentukan aturan pada sistem aplikasi untuk pembuatan password. Dimana pengguna harus menggunakan password dengan kombinasi yang kuat	*Weak Password Policy*	Jangan menyimpan *file* sensitive pada *directory public* (htaccess, phpinfo, web.config, .git, backup dan lain sebagainya)	*Sensitive File Exposure*
Melakukan validasi dan sanitasi (filter spesial karakter) inputan pengguna. Dan hal tersebut diterapkan tidak hanya di *front end* tetapi pada bagian back end sistem aplikasi.	*XSS*	Jika terdapat fitur *upload file*, pada fitur tersebut harus diterapkan aturan hanya mengizinkan ekstensi *file* tertentu sesuai kebutuhan. Ubah izin pada direktori *upload* sehingga *file* didalamnya tidak dapat dieksekusi. Jika memungkinkan, ubah nama *file* yang di*upload* menjadi random name agar menyulitkan penyerang pada saat akan mengeksekusi atau menjalan *file* yang telah di *upload* olehnya.	*Unrestricted File Upload*
Pada saat membangun sistem aplikasi, gunakan teknologi/*library* terbaru yang tidak memiliki *security issue*		Validasi dan sanitasi masukan atau nilai header yang dikirimkan tiap *requestnya*	*Host Header Injection*
Melakukan pemeriksaan dan validasi/sanitasi (filter spesial karakter) inputan. Hal tersebut diterapkan tidak hanya di sisi *front end* tetapi pada bagian *back end* juga	*HTML Injection*	Sistem harus memastikan bahwa data yang diinputkan sesuai dengan parameter	*No Bussiness Logic Data Validation*

Berdasarkan pedoman ancaman yang sudah dibuat, maka dapat dirumuskan menjadi daftar ceklis pedoman pembangunan dan pengembangan aplikasi yang bisa diimplementasikan sebaai berikut :

Table 3. Daftar ceklis pengamanan aplikasi

No.	Daftar Ceklis Pengamanan Aplikasi
1	Validasi inputan pengguna, terkhusus *special character* (‘”;)
2	Lakukan *encode* pada inputan pengguna
3	Batasi inputan pengguna (jika memungkinkan)
4	Jangan gunakan *query* mentah pada *coding* sistem
5	Validasi dan sanitasi inputan pengguna. Validasi dilakukan pada sisi FE dan BE
6	Validasi dan sanitasi inputan pengguna
7	Dapat juga melakukan *encode* inputan pengguna, jika pengguna menginputkan tag/code HTML
8	Terapkan control akses pada masing-masing pengguna. Pastikan hanya pengguna yang berwenang yang dapat melihat data atau menjalankan fungsi yang dimaksudkan
9	Terapkan *rate limit* pada *Backend*
10	Terapkan *captcha* dengan pengimplementasian yang benar
11	Menerapkan *header anti clickjacking* setiap *request*. Contohnya *X-Frame-Options*
12	Tambahkan *rule* pada *Backend system* untuk membuat password setidaknya dengan kombinasi huruf kapital, kecil, angka atau dan symbol dengan minimal 8 karakter.
13	Terapkan dan tetapkan semua outputan, termasuk pesan kesalahan
14	Menghapus dependensi, fitur, komponen dan dokumentasi yang tidak digunakan
15	*Update library* yang digunakan menjadi versi yang terbaru dan pastikan tidak ada *security issues*
16	Menerapkan mekanisme untuk memastikan bahwa jalur yang dikanonikalisasi dimulai dengan direktori dasar yang diharapkan
17	Validasi inputan pengguna
18	Jangan menyimpan *file sensitive* pada *directory public* (*htaccess, phpinfo, web.config*)
19	Hanya mengizinkan ekstensi *file* tertentu
20	Ubah izin pada direktori *upload* sehingga *file* didalamnya tidak dapat dieksekusi
21	Jika memungkinkan, ubah nama *file* yang di*upload* menjadi *random name*
22	Validasi dan sanitasi masukan atau nilai header yang dikirimkan tiap *request*

4. Kesimpulan

Berdasarkan hasil analisis kerentanan dari aplikasi yang telah dilakukan pengujian celah keamanannya, telah didapat usulan standarisasi pembuatan atau pengembangan sebuah aplikasi. Penulis berharap bahwa hasil analisis *threat modelling* yang dihasilkan bisa menjadi dasar untuk pembuatan dan pengembangan aplikasi dilingkungan Pemerintah Kota Bandug untuk dasar dalam pembuatan dan pengembangan aplikasi sehingga dapat meningkatkan keamanan dari aplikasi yang dibangun. Penulis juga berharap semoga kedepannya hasil yang sudah didapat bisa dikembangkan lagi dan dilakukan penganalisaan kembali guna untuk bisa melakukan *update* kebutuhan Teknik pengamanan sesuai dengan kondisi perkembangan keamanan teknonogi informasi dan komunikasi yang ada.

Daftar Pustaka

[1] Peraturan Presiden Republik Indonesia Nomor 95 Tahun 2018 tentang Sistem Pemerintahan Berbasis Elektronik.
[2] Kementerian Pendayagunaan Aparatur Negara dan Reformasi Birokrasi, "Panduan Penyelenggaraan Sistem Pemerintahan Berbasis Elektronik," 2018.
[3] Kementerian Komunikasi dan Informatika Republik Indonesia, "Peraturan Menteri Komunikasi dan Informatika Nomor 5 Tahun 2015 tentang Registrasi Penyelenggara Sistem Elektronik," 2015.
[4] Y. Liu, A. Sarabi, J. Zhang, P. Naghizadeh, M. Kavousi, and M. Liu, "Cloudy with a Chance of Breach: Forecasting Cyber Security Incidents," in 24th USENIX Security Symposium (USENIX Security 15), 2015, pp. 1009–1024.
[5] A. Sharma and S. K. Sahay, "Threats and Vulnerability Assessment in Cyber Security," in International Conference on Computing, Communication and Automation (ICCCA), 2017, pp. 1122–1126.
[6] M. Wazid, A. K. Das, N. Kumar, A. V. Vasilakos, and J. J. P. C. Rodrigues, "Design and Analysis of Secure Lightweight Remote User Authentication and Key Agreement Scheme in Internet of Drones Deployment," IEEE Internet of Things Journal, vol. 6, no. 2, pp. 3572–3584, 2019.
[7] A. Arora, S. Bansal, and S. Kandpal, "A Secure *File* Storage System Using Blockchain and Cryptography," in 2019 9th International Conference on Cloud Computing, Data Science & Engineering (Confluence), 2019, pp. 543–548.
[8] J. W. Creswell and J. D. Creswell, Research Design: Qualitative, Quantitative, and Mixed Methods Approaches, 5th ed. Sage Publications, 2018.
[9] S. K. Ningrum, “Implementasi *Scrum* Pada Manajemen Proyek Pengembangan Perangkat Lunak Pemesan Undangan (Studi Kasus: Paperlust),” Skripsi, Fakultas Teknologi Industri., Universitas Islam Indonesia, 2020.
[10] M. Rizky and Y. Sugiarti, " Pengunaan Metode *Scrum* Dalam Pengembangan Perangkat Lunak: Literature Review," Journal of Computer Science an Engineering (JCSE), 2021, pp. 41–48.
[11] IEEE Std 829.1-2021 - IEEE Standard for Application of *Scrum* Framework.
[12] IEEE Std 16326-2019 - IEEE Standard for Software Project Management Processes.
[13] IEEE Std 2675-2017 - IEEE Recommended Practice for the Adoption of Agile.
[14] IEEE Std 1074-2006 - IEEE Standard for Developing Software Life Cycle Processes.
[15] IEEE Std 27034-1-2019 - ISO/IEC/IEEE International Standard - Application Security - Part 1: Overview and Concepts.