 ISSN : 2503-3212

Keamanan Berbasis Service Oriented Architecture Menggunakan Oauth 2.0 dan Json Web Token

Thomas Adi Nugroho[1], Asep Id Hadiana [2], Sigit Anggoro[3]

Program Studi Informatika, Fakultas Sains dan Informatika Universitas Jenderal Achmad Yani
Jl.Terusan Jenderal Sudirman, 148 Cimahi, Jawa Barat, Indonesia *Email: thomasadin18@if.unjani.ac.id

ABSTRACT

Digital transformation has been proven to bring many benefits to company growth. In today's digital era, many companies want to generate business innovation and make it happen through applications. To achieve this, companies need solutions that can make teams agile and focused on teamwork, and create to deliver value to customers. based on these needs makes services oriented architecture a fast go-to solution for a company, this is in line with previous research which shows that the use of services oriented architecture is better than monolithic architecture because the system built is broken down into smaller parts and interconnected through Restful API, but the use of microservices architecture brings another challenge, namely on security issues. Because of these problems, this research requires an information security strategy where a layer of security defense is placed on the system. Authorization and Authentication are strategies that are suitable for implementation on microservices. Authorization in this study uses JSON Web Token and OAuth 2.0 on the authentication side. Based on Penetration Testing that has been carried out, microservices that use JSON Web Token and OAuth 2.0 security show 5 security holes that are tracked to be medium and low.

Keywords*: Services Oriented Architecture, Authentication, Authorization, Penetration testing*

Abstrak

Transformasi digital telah terbukti membawa banyak manfaat bagi pertumbuhan perusahaan. Di era digital saat ini, banyak perusahaan ingin menghasilkan inovasi bisnis dan mewujudkannya melalui aplikasi. Untuk mencapai hal itu, perusahaan memerlukan solusi yang dapat membuat tim menjadi agile dan fokus pada kerja sama tim, serta berinovasi untuk memberikan value kepada customer. berdasarkan kebutuhan tersebut menjadikan services oriented architecture sebagai fast go-to solution bagi suatu perusahaan, hal ini sejalan dengan penelitian terdahulu yang menunjukan bahwa penggunaan services oriented architecture lebih baik dari pada monolithic architecture karena sistem yang dibangun dipecah menjadi bagian yang lebih kecil dan saling terhubung melalui Restful API, namun penggunaan microservices architecture membawa tantangan lain yaitu pada masalah keamanan. Oleh karena permasalahan tersebut maka penelitian ini memerlukan sebuah strategi keamanan informasi dimana lapisan pertahanan keamanan ditempatkan pada system. autorisasi dan Authentikasi merupakan strategi yang cocok untuk diimpelementasikan pada microservices, Autorisasi pada penelitian ini menggunakan JSON Web Token serta OAuth 2.0 pada sisi authentication Berdasarkan Pengujian penetration testing yang telah dilakukan, microservices yang menggunakan keamanan JSON Web Token dan OAuth 2.0 menunjukan 5 celah keamanan yang dikategorikan menjadi medium, dan low.

Kata Kunci : *Services Oriented Architecture, Authentication, Autorisasi, Penetration testing*

PENDAHULUAN

Transformasi digital telah membuktikan dirinya membawa banyak manfaat bagi pertumbuhan perusahaan. Di era digital saat ini, banyak perusahaan ingin menghasilkan inovasi bisnis dan mewujudkannya melalui aplikasi. Untuk mencapai hal itu, perusahaan memerlukan solusi yang dapat membuat tim menjadi *agile* dan fokus pada kerja sama tim, serta berinovasi untuk memberikan *value* kepada *customer*. Dalam beberapa tahun terakhir, Service-Oriented Architecture (SOA) telah mendapat perhatian yang meningkat sejalan dengan bergerak menuju mengatasi tantangan yang terkait dengan perbaikan dan pemeliharaan

lingkungan yang beragam. Untuk memodernisasi sistem perangkat lunak organisasi, bermigrasi dari sistem lama ke sistem berbasis SOA telah menjadi tren utama[1].

Ini karena SOA menawarkan integrasi yang fleksibel dan layanan dapat digunakan kembali karena arsitektur modular berbasis layanannya, SOA juga menawarkan transparansi karena merangkum beberapa aplikasi dan sumber data dalam bentuk black box. Dengan cara ini, kumpulan sumber daya Teknologi Informasi (TI) yang terintegrasi tetap dapat diakses meskipun terdapat beragam teknologi, kode bahasa, fungsionalitas, dan platform[2]. Berkenaan dengan sektor industri, sejauh ini, SOA telah terbukti menjadi paradigma kunci di berbagai industri seperti perbankan, kesehatan, transportasi, dll. Seiring dengan banyak manfaatnya, beberapa studi utama yang teridentifikasi mengungkapkan bahwa organisasi tidak dapat mewujudkannya. manfaat penuh dari adopsi SOA karena beberapa alasan. Namun, tidak ada studi sistematis terperinci yang berbagi kriteria penting yang berpengaruh untuk adopsi dan implementasi SOA yang berhasil. Untuk mengisi kesenjangan ini, sangat penting untuk menyelidiki faktor signifikan adopsi SOA dalam organisasi karena pemahaman tentang faktor-faktor ini akan membantu organisasi memaksimalkan manfaat implementasi SOA [3].

Keamanan jaringan merupakan tantangan lain SOA, meskipun mengamankan SOA lebih sulit daripada *monolithic*, namun dapat dilindungi secara efektif dengan menetapkan strategi dan mengikuti praktik terbaik [4]. Adanya contoh insiden keamanan pada infrastruktur keamanan Indonesia, yang terjadi pada eHAC, dimana terdapat 1.3 juta data pengguna eHAC beresiko di salahgunakan pada bulan juli tahun 2021 [5].

Beberapa protokol bekerja untuk menjaga keamanan. Salah satu protokol yang bertujuan untuk mengautentikasi pihak ketiga aplikasi dan mengizinkan mereka untuk mengakses data pengguna dalam cara dikendalikan adalah OAuth (Otorisasi Terbuka) [6]. Diperkenalkan pada tahun 2009, OAuth telah diterima secara luas di waktu yang singkat dan dapat disebut sebagai standar de facto untuk otorisasi. OAuth memungkinkan pengguna untuk menyediakan pihak ketiga aplikasi akses ke sumber daya yang dilindungi pengguna disimpan di beberapa server tanpa membocorkan kata sandi pengguna atau lainnya kredensial rahasia secara transparan. Protokol ini memiliki melewati dua versi utama, dengan versi terbaru OAuth 2.0 tidak kompatibel dengan pendahulunya OAuth 1.0 [7].

Token Web JSON (JWT) adalah objek JSON yang didefinisikan dalam RFC 7519 sebagai metode komunikasi yang aman antara dua pihak. JWT telah digunakan dalam beberapa penelitian yang disajikan dalam literatur untuk mempertahankan otentikasi klien saat berinteraksi dengan server. Aplikasi Cloud SaaS dan aplikasi manajemen smartphone menggunakan mekanisme JWT untuk mengautentikasi klien untuk menggunakan sumber daya server atau mengakses perangkat platform IoT. Pendekatan ini memiliki beberapa kerentanan asli yang dapat disalahgunakan oleh penyerang untuk mendapatkan sumber daya server dalam jangka waktu yang lama. Kelemahan umum dengan pendekatan ini adalah bahwa token yang sama digunakan hingga JWT kedaluwarsa atau pengguna logout. [8]

Dalam penelitian sebelumnya meng konfigurasikan ulang antara fleksibility dan interoperabilitas. Rest adalah arsitektur berorientasi layanan yang berjalan pada suatu protocol HTTP, yang belum tentu lebih cepat dari dari protokol SOAP atau singkatan dari Subjective (Subjektif), Objective (Objektif), Assesment (Penilaian), dan Plan (Perencanaan). Oleh karena itu solusi yang diusulkan oleh kertas komentar mungkin tidak sesuai untuk fleksibilitas dan interoperabilitas tingkat perangkat. Pada saran yang ditulis oleh peneliti tersebut mengungkapkan bahwa mengembangkan model formal yang lebih rinci. Oleh karena itu peneliti akan memberikan metode keamanan yang lebih rinci ke arah arsitektur keamanan dengan menggunakan Oauth 2.0 dan Json Web Token.

Dalam melakukan pencegahan agar perangkat lunak tidak dapat di tembus oleh serangan siber, di perlukan protokol tentang otorisasi dan juga authentikasi data penguna dalam mengakses perangkat lunak. Dengan Menggunakan OAuth 2.0 sebagai protokol keamanan untuk otorisasi dan JSon Web Token sebagai metode keamanan autentikasi untuk menjaga keaslian data dalam bentuk token. Oleh karena itu penelitian ini bertujuan untuk mengamankan layanan *Service oriented architecture* menggunakan OAuth 2.0 dan JSon Web Token dari serangan siber.

METODE PENELITIAN

1. **Tahap Pertama** merupakan *studi literature* pengumpulan, tinjauan pustaka terhadap konsep dan metode kerja yang digunakan untuk memecahkan masalah yang diangkat dalam penelitian ini.
2. **Tahap Kedua** merupakan tahap analisis dan perancangan perangkat lunak. Pada tahap ini dilakukan analisis kebutuhan fungsional pada aplikasi. Untuk memudahkan proses perancangan perangkat lunak, penelitian ini akan menggunakan diagram UML (*Unified Modeling Language*) untuk memodelkan fungsi fungsi yang ada pada sistem
3. **Tahap Ketiga** merupakan tahap implementasi arsitektur *microservice* pada aplikasi. Pada tahap ini dilakukan pembangunan menggunakan *framework Express* JS untuk bagian *backend* dan *next JS* untuk bagian *frontend*, serta Implementasi keamanan dengan menggunakan *JSON Web Token (JWT)*
4. **Tahap Keempat** merupakan Pengujian keamanan menggunakan *penetration testing, Penetration testing* adalah proses *ethical hacking* yang melibatkan penilaian aplikasi atau infrastruktur organisasi untuk berbagai jenis kerentanan (*vulnerabilities*). Didalam penetration testing terdapat beberapa langkah dalam pengujian, seperti *planning test*, eksekusi Pengujian dan *reporting*. Proses *penetration testing* ini membantu untuk mengeksploitasi berbagai kerentanan dalam sistem.

Gambar 1.1 Metode Penelitian

HASIL DAN PEMBAHASAN

Tujuan yang ingin dicapai dari penelitian ini adalah untuk menerapkan kerangka keamanan *JSon Web Token* dan *OAuth 2.0* dalam suatu keamanan *service oriented architecture*, lalu melakukan simulasi serangan pada *service oriented architecture* untuk menemukan kelemahan dan kerentanan (*vulnerability*) dengan menggunakan *black box testing.*

1. Planning Test

Tabel 1 Penggunaan tools pengujian

No	Tools	Kegunaan
1	Postman	Postman mengirim permintaan API ke server web dan menerima respons, apa pun itu. Tidak diperlukan pekerjaan tambahan atau konfigurasi kerangka kerja saat mengirim dan menerima permintaan di Postman. Pengujian menggunakan aplikasi Postman dengan tiga metode bawaan yaitu GET, POST dan PUT.
2	*OWASP Zap*	*OWASP Zed Attack Proxy (Open Web Application Security Project ZAP)* adalah sebuah aplikasi yang digunakan untuk melakukan *penetration testing* guna menemukan *vulnerabilities* dalam suatu *web applications*.

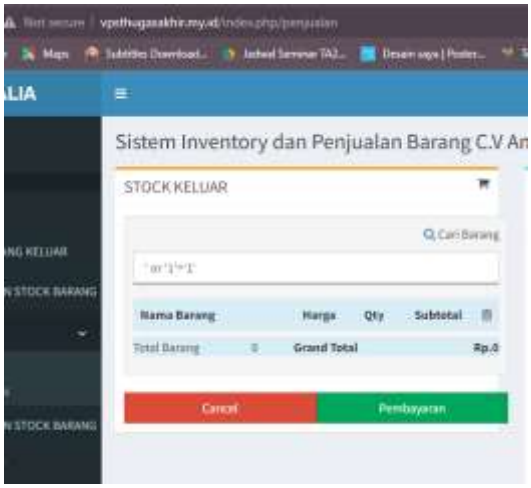
Tabel 2 Serangan yang akan di pakai untuk melakukan serangan

No	Jenis Serangan	Kegunaan
1	SQL INJECTION	Serangan injeksi SQL memungkinkan penyerang untuk memalsukan identitas, mengutak-atik data yang ada, menyebabkan masalah penolakan seperti membatalkan transaksi atau mengubah saldo, memungkinkan pengungkapan lengkap semua data pada sistem, menghancurkan data atau membuatnya tidak tersedia, dan menjadi administrator system dari server basis data.

2. Eksekusi Serangan

Pada serangan ini menggunakan serangan SQL INJECTION, target yang akan di serang ada pada fitur penjualan yang menampilkan data data yang sudah di jual. Berikut pengujian nya yaitu memasukan sql injection cheat sheet dengan code ‘ or ‘1’=’1’ pada kolom pencarian yang berfungsi menambahkan isi query dari pemanggilan data penjualan dari basis data. Serangan ini di tujukan untuk merubah isi query dari pemanggilan data penjualan yang tersimpan pada database.

Gambar 1. Melakukan SQL Injection



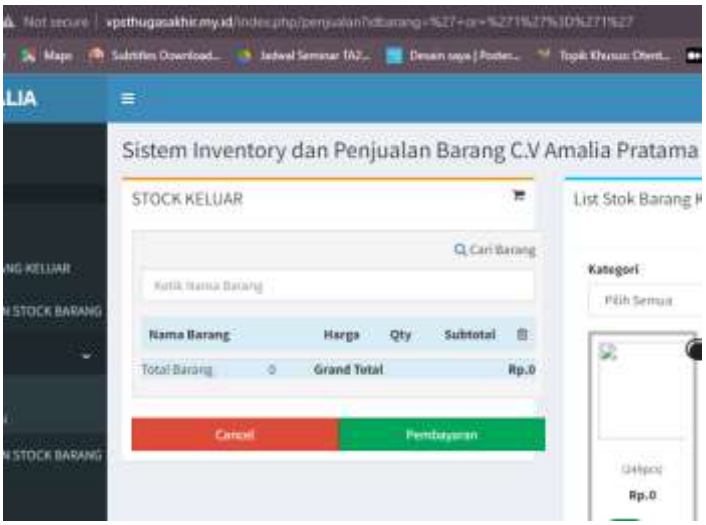
Berikut adalah query yang dituju dari kolom pencarian yang sudah di masukan code sql injection. Disini code ‘ or ‘1’=’1’ akan mengisi query dari ‘q’.

Gambar 2. Query yang akan diserang oleh SQL Injection

```
function caribarang()
{
    $key = $this->input->get('q');
    $data = $this->Model_penjualan->hasilcari($key);
    foreach ($data as $result) {
        echo "<a href='" . base_url() . 'index.php/Penjualan/tambah_barang/' . $result->id_barang . "/1'>" . $result->nama_barang . "</a>
    }
}
```

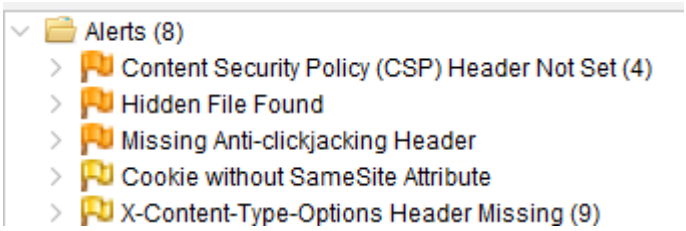
Hasil dari jenis serangan ini adalah menampilkan barang barang yang sudah terjual yang tersimpan pada basis data. Namun code di atas gagal menampilkan data penjualan.

Gambar 3. Hasil dari Serangan SQL Injection



3. Ekseskusi Pengujian
   Dalam penggunaan aplikasi OWASP Zap dalam melakukan pencarian celah keamanan, dengan memasukan url website seperti http://vpsthugasakhir.my.id/auth/login pada menu *automated scan*, dan mengklik *button attack*.

Gambar 4. Hasil pengujian oleh OWASP ZAP



Setelah dilakukan proses *attack.* Maka akan menunjukan daftar celah keamanan pada url website tersebut pada gambar 4. 22

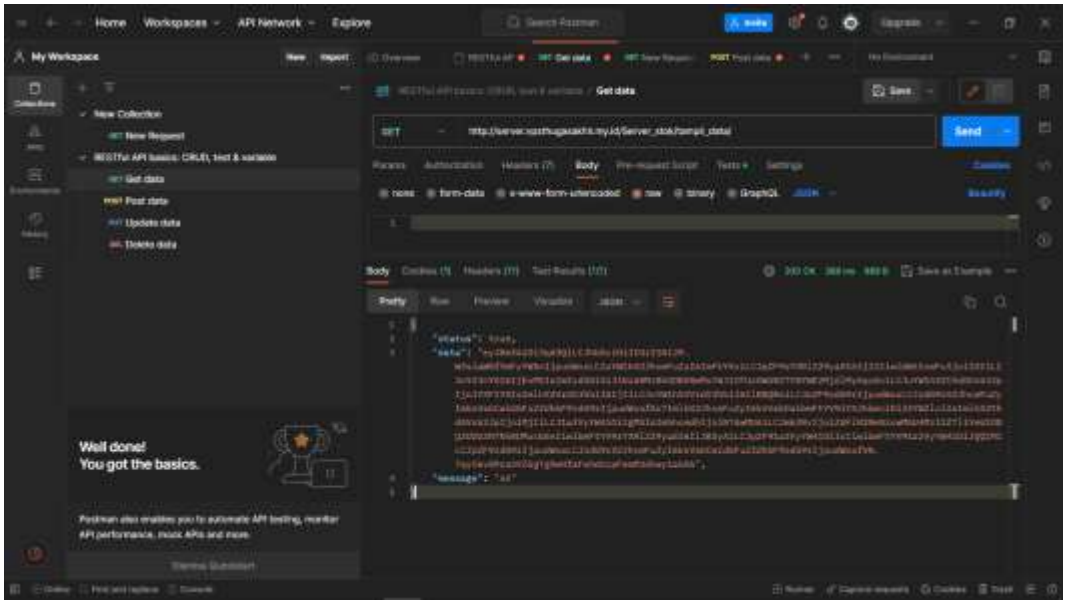
Tabel 3. Hasil Pengujian

Domain	Risk		
	High	Medium	Low
http://vpsthugasakhir.my.id/auth/login	0	3	2

Pada hasil pengujian dengan menggunakan tools OWASP Zap, menunjukan bahwa domain url http://vpsthugasakhir.my.id/auth/login yang memakai keamanan Oauth 2.0 hanya terdapat 5 kerentanan.

Kemudian dalam pengujian menggunakan postman menggunakan 2 jenis url yang sudah ter enkripsi dan urlyang bisa di lihat oleh admin seperti di bawah ini

Gambar 5. Hasil Pengujian menggunakan POSTMAN



KESIMPULAN

Pada bagian akhir ini, peneliti akan memberikan beberapa kesimpulan berdasarkan hasil penelitian yang menguji kerentanan dari pemakaian OAuth 2.0 dan Json Web Token pada Service-Oriented Architecture melalui beberapa pengujian. Terdapat 2 jenis pengaman untuk Service-Oriented Architecture (SOA) ini yaitu OAuth 2.0 dan Json Web Token. OAuth 2.0 digunakan untuk proses login kepada Service-Oriented Architecture menggunakan akun google untuk pengecekan otentikasi sehingga tidak lagi memerlukan form

id dan password. Json Web Token digunakan untuk membuat token untuk mengenkripsi data sensitive yang ada pada Service-Oriented Architecture.

Sistem yang telah dibuat kemudian dilakukan pengujian dengan menggunakan *penetration testing,* serta menggunakan metode *black box testing*. Pada *penetration testing* dilakukan tahapan *planning test*, eksekusi pengujian, dan *reporting.* Hasil pengujian Pengujian *penetration testing* yang menggunakan tools OWASP Zap menunjukan bahwa domain url http://vpsthugasakhir.my.id/auth/login yang memakai keamanan OAuth 2.0 terdapat 5 kerentanan. Celah keamanan yang didapatkan lalu di evaluasi, pada tahapan *reporting* yaitu :

Vulnerability ID	Celah Keamanan	Risk
VULN-001	*Content Security Policy (CSP) Header Not Set*	Medium
VULN-002	*Hidden File Found*	Medium
VULN-003	*Missing Anti-clickjacking Header*	Medium
VULN-004	*Cookie without SameSite Attribute*	Low
VULN-005	*X-Content-Type-Options Header Missing*	Low

Dalam penelitian ini masih terdapat kekurangan yang belum disampaikan oleh penulis, maka dari itu untuk penelitian yang lebih baik lagi penulis menyarankan untuk:

a. Untuk meneliti celah keamanan tidak hanya pada bagian *authentication & authorization.*

b. *Penetration testing* menggunakan *tools* lain untuk perbandingan hasil pengujian.

DAFTAR PUSTAKA

[1] N. Niknejad, W. Ismail, I. Ghani, B. Nazari, M. Bahari, and A. R. B. C. Hussin, “Understanding Service-Oriented Architecture (SOA): A systematic literature review and directions for further investigation,” *Inf. Syst.*, vol. 91, p. 101491, 2020, doi: 10.1016/j.is.2020.101491.

[2] A. Kurniawan, A. A. Nugroho, and S. Mulyono, “Sistem Informasi Rental Mobil Terintegrasi Menggunakan Service Oriented Architecture,” *TRANSISTOR Elektro dan Inform.*, vol. 2, no. 2, pp. 134–142, 2018, [Online]. Available: http://jurnal.unissula.ac.id/index.php/EI/article/view/3053/2216.

[3] O. Somantri and I. D. Hasta, “Implementasi e-Government Pada Kelurahan Pesurungan Lor Kota Tegal Berbasis Service Oriented Architecture (SOA),” *J. Inform. Pengemb. IT*, vol. 2, no. 1, pp. 23–29, 2017.

[4] T. Yarygina and A. H. Bagge, “Overcoming Security Challenges in Microservice Architectures,” *Proc. - 12th IEEE Int. Symp. Serv. Syst. Eng. SOSE 2018 9th Int. Work. Jt. Cloud Comput. JCC 2018*, pp. 11–20, 2018, doi: 10.1109/SOSE.2018.00011.

[5] BBC, “Data eHAC bocor, pakar siber sebut ‘Infrastruktur keamanan digital pemerintah Indonesia sangat buruk,’” *bbc.com*, 2021. https://www.bbc.com/indonesia/indonesia-58406164 (accessed Oct. 14, 2022).

[6] M. P. Oauth, “Jurnal JARKOM Vol . 5 No . 2 Desember 2017 E- ISSN : 2338-6304 PERANCANGAN DAN IMPLEMENTASI SSO (SINGLE SIGN ON) Jurnal JARKOM Vol . 5 No . 2 Desember 2017 E- ISSN : 2338-6304,” vol. 5, no. 2, pp. 102–108, 2017.

[7] S. Pai, Y. Sharma, S. Kumar, R. M. Pai, and S. Singh, “Formal Verification of OAuth 2 . 0 using Alloy Framework,” pp. 655–659, 2011, doi: 10.1109/CSNT.2011.141.

[8] S. Ahmed, “An authentication based scheme for applications using JSON web token,” *2019 22nd Int. Multitopic Conf.*, pp. 1–6, 2019.

[9] V. Cardellini, E. Casalicchio, V. Grassi, S. Iannucci, F. Lo Presti, and R. Mirandola, “MOSES: A framework for qos driven runtime adaptation of service-oriented systems,” *IEEE Trans. Softw. Eng.*, vol. 38, no. 5, pp. 1138–1159, 2012, doi: 10.1109/TSE.2011.68.

[10] K. Avila, P. Sanmartin, D. Jabba, and M. Jimeno, “Applications based on service-oriented architecture (SOA) in the field of home healthcare,” *Sensors (Switzerland)*, vol. 17, no. 8, 2017, doi: 10.3390/s17081703.

[11] E. MacLennan and J. P. Van Belle, “Factors affecting the organizational adoption of service-oriented architecture (SOA),” *Inf. Syst. E-bus. Manag.*, vol. 12, no. 1, pp. 71–100, 2014, doi: 10.1007/s10257-012-0212-x.
[12] Sam Newman, *Building Microservices*. O’Reilly Media, Inc., 2015.
[13] M. Villamizar *et al.*, “Cost comparison of running web applications in the cloud using monolithic, microservice, and AWS Lambda architectures,” *Serv. Oriented Comput. Appl.*, vol. 11, no. 2, pp. 233–247, 2017, doi: 10.1007/s11761-017-0208-y.
[14] A. M. Tisnadinata, “Komunikasi Antar Service Dalam Arsitektur Microservices,” 2020. https://medium.com/komunikasi-antar-service-dalam-arsitektur/komunikasi-antar-service-dalam-arsitektur-microservices-8dcf316ca49e (accessed Jan. 04, 2023).
[15] Imperva, “Information Security: The Ultimate Guide,” 2020. https://www.imperva.com/learn/data-security/information-security-infosec/#:~:text=Information security protects sensitive information,financial data or intellectual property (accessed Jan. 04, 2023).
[16] and T. C. M. Trnka, A. S. Abdelfattah, A. Shrestha, M. Coffey, “Systematic Review of Authentication and Authorization Advancements for the Internet of Things,” vol. 22, p. 4, 2022, doi: 10.3390/s22041361.
[17] Auth0, “Introduction to JSON Web Tokens.” https://jwt.io/introduction (accessed Jan. 05, 2023).
[18] Auth0, “What is OAuth 2.0?” https://auth0.com/intro-to-iam/what-is-oauth-2 (accessed Jan. 09, 2023).
[19] A. Alanda, D. Satria, H. A. Mooduto, and B. Kurniawan, “Mobile Application Security Penetration Testing Based on OWASP,” *IOP Conf. Ser. Mater. Sci. Eng.*, vol. 846, no. 1, 2020, doi: 10.1088/1757-899X/846/1/012036.
[20] M. Touseef, N. Anwer, A. Hussain, and A. Nadeem, “Testing from UML Design using Activity Diagram: A Comparison of Techniques,” *Int. J. Comput. Appl.*, vol. 131, no. 5, pp. 41–47, 2015, doi: 10.5120/ijca2015907354.
[21] N. A. Zafar, “Formal Specification and Verification of Few Combined Fragments of UML Sequence Diagram,” *Arab. J. Sci. Eng.*, vol. 41, no. 8, pp. 2975–2986, 2016, doi: 10.1007/s13369-015-1999-9.
[22] P. K. Arora and R. Bhatia, “Agent-Based Regression Test Case Generation using Class Diagram, Use cases and Activity Diagram,” *Procedia Comput. Sci.*, vol. 125, pp. 747–753, 2018, doi: 10.1016/j.procs.2017.12.096.