CyberSecurity dan Forensik Digital e-ISSN: 2615-8442
Vol. 9, No. 1, Mei 2026, hlm. 54-65

Mitigasi Insider Threats Menggunakan Zero Trust Architecture (NIST SP 800-207) Pada Aplikasi Web

Andy Victor Pakpahan[1], Aldiansyah Reksa Pratama Wicaksono[2]

[1,2]Institut Digital Ekonomi LPKIA, Bandung, Indonesia
Email: [1]abang@lpkia.ac.id, [2]reksapratamawicaksono@gmail.com

Abstrak

Penerapan keamanan tradisional berbasis *perimeter* saat ini tidak lagi memadai untuk ancaman *internal* seperti *lateral movement* dan eskalasi hak akses. Hal ini disebabkan oleh model keamanan konvensional yang cenderung memberikan kepercayaan penuh terhadap entitas yang sudah berada di dalam jaringan. Penelitian ini bertujuan mengimplementasikan *Zero Trust Architecture* (ZTA) berbasis standar NIST SP 800-207 pada aplikasi *web* Laravel untuk meningkatkan kontrol akses. Metodologi yang digunakan meliputi pemodelan komponen ZTA (*Policy Engine*, *Policy Administrator*, dan *Policy Enforcement Point*) melalui integrasi *Multi-Factor Authentication* (MFA), *Role-Based Access Control* (RBAC), dan pencatatan *log* aktivitas. Proses pengembangan juga melibatkan konfigurasi *middleware* khusus pada Laravel untuk memastikan setiap permintaan akses diverifikasi secara ketat. Selain itu dilakukan simulasi serangan *lateral movement* dan *privilege escalation* untuk menguji ketahanan sistem. Hasil pengujian menunjukkan bahwa arsitektur yang dibangun mampu membatasi akses secara ketat berdasarkan identitas dan peran, serta berhasil memitigasi upaya pergerakan *lateral* dalam aplikasi. Simpulan dari penelitian ini menegaskan bahwa pendekatan "*never trust, always verify*" efektif dalam memperkuat keamanan aplikasi *web*, meskipun implementasi algoritma kepercayaan dinamis masih memerlukan pengembangan lebih lanjut sebagai kontribusi masa depan.

Kata kunci: *Zero Trust Architecture*, NIST SP 800-207, Laravel, *Lateral Movement*, Keamanan Aplikasi *Web*

Mitigating Insider Threats Using Zero Trust Architecture (NIST SP 800-207) In Web Applications

Abstract

Traditional perimeter-based security applications are no longer sufficient to mitigate internal threats such as lateral movement and privilege escalation. This is due to conventional security models that tend to grant implied trust to entities already within the network. This study aims to implement Zero Trust Architecture (ZTA) based on the NIST SP 800-207 standard on a Laravel web application to enhance access control. The methodology involves modeling core ZTA components (Policy Engine, Policy Administrator, and Policy Enforcement Point) through the integration of Multi-Factor Authentication (MFA), Role-Based Access Control (RBAC), and comprehensive activity logging. The development process also involves configuring custom middleware in Laravel to ensure every access request is strictly verified. Furthermore, simulations of lateral movement and privilege escalation attacks were conducted to test system resilience. The results indicate that the constructed architecture is capable of strictly limiting access based on identity and roles, effectively mitigating lateral movement attempts within the application. This study concludes that the "never trust, always verify" approach is effective in strengthening web application security, although the implementation of dynamic trust algorithms remains a necessity for future development.

***Keywords**: Zero Trust Architecture,* NIST SP 800-207*,* Laravel, *Lateral Movement, Web Application Security*

1. PENDAHULUAN

Perkembangan teknologi informasi telah mendorong digitalisasi di berbagai sektor (Edo et al., 2022), namun juga meningkatkan ancaman dalam bidang keamanan siber (*cybersecurity*) (Y. Li & Liu, 2021), terutama terhadap data dan sistem digital (Ray, 2023). Ancaman ini muncul di berbagai lapisan, mulai dari serangan injeksi dan *malware* di tingkat aplikasi, *Distributed Denial of Service* dan interupsi di tingkat jaringan, hingga *ransomware* dan *insider threats* yang mengancam sistem secara menyeluruh (Kumari & Jain, 2023). Selain itu, ancaman dari penyerang yang berhasil menembus lapisan pertahanan luar, kemudian dapat bergerak secara leluasa di dalam jaringan internal (Aslan et al., 2023), sebuah ancaman yang dikenal sebagai pergerakan *lateral* (*lateral movement*) (Ge & Zhu, 2024). Untuk

menghadapinya, banyak organisasi masih mengandalkan model keamanan tradisional berbasis *perimeter* (Villegas-Ch & García-Ortiz, 2023), yang memisahkan jaringan *internal* dan eksternal melalui mekanisme seperti *Firewall* atau *VPN (Abdulazeez et al., 2020)*. Sayangnya, pendekatan ini memiliki kelemahan mendasar (Rafi et al., 2025), karena entitas yang berhasil melewati autentikasi awal sering kali diberi kepercayaan penuh tanpa verifikasi lanjutan (Azad et al., 2024), membuka celah bagi eksploitasi dari dalam.

Aplikasi *web* kini menjadi komponen krusial dalam sistem informasi *modern* dan layanan digital di berbagai sektor. *Website* tidak hanya berfungsi sebagai antarmuka layanan, tetapi juga sebagai tempat penyimpanan data sensitif, media komunikasi, dan sarana transaksi dalam jaringan (Herrada-Lores et al., 2022). Pada *platform* pendaftaran dan manajemen data publik, risiko keamanan mencakup tidak hanya kerahasiaan data, tetapi juga integritas dan ketersediaan layanan (Elanda & Buana, 2020). Sifatnya yang terbuka dan dapat diakses dari berbagai lokasi dan perangkat, menjadikan aplikasi *web* sebagai salah satu target utama serangan siber (Fronita, 2023).

Berbagai studi menunjukkan bahwa aplikasi *web* memiliki celah keamanan kritis yang sering dimanfaatkan penyerang (Alwi et al., 2020). Berdasarkan *Open Web Application Security Project* (OWASP) Top 10 (Priyawati et al., 2022), kerentanan seperti *broken authentication*, *insufficient access control*, dan *security misconfiguration* masih mendominasi insiden keamanan (Glemser, 2022; J. Li, 2020). Celah ini biasanya disebabkan oleh lemahnya sistem autentikasi, pengelolaan hak akses, dan konfigurasi yang tidak tepat (Kumi et al., 2021).

Berbagai solusi keamanan telah banyak diterapkan untuk melindungi aplikasi *web*, seperti penggunaan *Web Application Firewall* (WAF), enkripsi data, *token Cross-Site Request Forgery* (CSRF) (Sadeghi & Hadavi, 2021), protokol *HyperText Transfer Protocol Secure* (HTTPS) (Prayama et al., 2021), serta autentikasi dua faktor (MFA) (Zaky & Saxe, 2022). *Zero Trust Architecture* (ZTA) hadir sebagai pendekatan komprehensif untuk menjawab tantangan ini. Meskipun awalnya dikembangkan dalam konteks *cloud* dan *Internet of Thing* (IoT) (Azad et al., 2024; Pakpahan, 2024), prinsip-prinsipnya sangat relevan untuk keamanan aplikasi *web (Hu et al., 2022)*. ZTA mengasumsikan bahwa tidak ada entitas yang dapat dipercaya tanpa proses verifikasi berkelanjutan, dan setiap akses dikendalikan secara ketat dengan prinsip *least privilege (Buck et al., 2021; Rose et al., 2020)*. Meskipun prinsip ini bersifat universal, penerapannya pada lapisan aplikasi (*application-level*) memiliki karakteristik berbeda dibandingkan pada lapisan jaringan, terutama dalam hal manajemen sesi dan otorisasi dinamis yang langsung bersentuhan dengan logika bisnis. Beberapa studi literatur, seperti yang dikemukakan oleh (Phiayura & Teerakanok, 2023; Teerakanok et al., 2021), telah mengidentifikasi bahwa hambatan utama adopsi *Zero Trust* adalah kompleksitas transisi dari sistem tradisional. Namun, masih terdapat kelangkaan literatur yang membahas pemetaan teknis standar NIST SP 800-207 secara spesifik ke dalam kontrol logika aplikasi *web* menggunakan *framework modern*, yang menjadi fokus utama dalam penelitian ini.

Meskipun prinsip ZTA mulai banyak diadopsi, implementasinya pada aplikasi *web* skala kecil hingga menengah masih menghadapi tantangan berupa kompleksitas konfigurasi dan keterbatasan infrastruktur (Pradeep Nayak et al., 2022; Sarkar et al., 2022; Teerakanok et al., 2021). Oleh karena itu, penelitian ini bertujuan mendemonstrasikan penerapan ZTA berbasis NIST SP 800-207 melalui integrasi MFA, RBAC, dan *log* aktivitas yang dirancang dalam tiga komponen utama: *Policy Engine* (PE), *Policy Administrator* (PA), dan *Policy Enforcement Point* (PEP) (Campbell, 2020; Laverdière et al., 2021; Shore et al., 2021). Penerapan pada *level* aplikasi ini memiliki limitasi dimana ia berfokus pada kontrol logika bisnis dan integritas sesi pengguna, namun tidak mencakup keamanan infrastruktur jaringan secara menyeluruh. Sebagai *platform* pembuktian, *framework* Laravel dipilih karena keunggulan ekosistemnya, seperti paket Spatie serta sistem *Guard* dan *Service Container* yang memudahkan validasi identitas secara terpusat (Mary & Febriyani, 2025; Putra et al., 2025). Meskipun *framework* lain memiliki fitur serupa, Laravel dipilih sebagai strategi pragmatis untuk menghasilkan prototipe ZTA yang ringan, mudah direplikasi, dan cocok untuk organisasi dengan sumber daya terbatas tanpa harus merombak seluruh infrastruktur jaringan bawah. Strategi ini memastikan bahwa setiap permintaan (*request*) tetap divalidasi secara berkelanjutan sesuai prinsip dasar ZTA tanpa mengorbankan skalabilitas pengembangan aplikasi.

1.1. *Zero Trust Architecture* (ZTA)

Zero Trust Architecture (ZTA) merupakan paradigma keamanan informasi yang mengasumsikan bahwa tidak ada entitas yang dapat dipercaya, baik yang berasal dari dalam maupun luar jaringan. Pendekatan ini menggantikan model keamanan tradisional berbasis perimeter yang sudah tidak efektif menghadapi serangan *modern*. Dalam arsitektur *Zero Trust*, setiap permintaan akses harus melalui proses autentikasi dan otorisasi yang ketat dan dinamis. Dokumen NIST SP 800-207 menjadi referensi utama dalam pengembangan arsitektur ini, dengan menekankan konsep seperti autentikasi berkelanjutan, kontrol akses berbasis peran, serta prinsip *least privilege*.

1.2. Komponen *Zero Trust Architecture* (NIST SP 800-207)

Arsitektur logis *Zero Trust* mencakup tiga komponen utama (Rose et al., 2020):

1 *Policy Engine* (PE): Komponen yang membuat keputusan akses berdasarkan kebijakan organisasi.
2 *Policy Administrator* (PA): Mengelola proses pemberian akses dari PE ke sumber daya.
3 *Policy Enforcement Point* (PEP): Titik kendali yang menegakkan keputusan akses terhadap permintaan pengguna.

1.3. *Multi-Factor Authentication* (MFA), *Least Privilege,* Eskalasi Hak Akses, dan Akses Kontrol

MFA adalah metode autentikasi yang menggunakan kombinasi dari dua atau lebih faktor: pengetahuan (*password*), kepemilikan (*token*), dan biometrik. Penggunaan MFA membantu mengurangi risiko serangan seperti pencurian kredensial dan *phishing* (Suleski et al., 2023).

Prinsip *least privilege* merupakan konsep fundamental dalam keamanan sistem informasi yang menyatakan bahwa setiap entitas (pengguna, proses, sistem) hanya diberikan hak akses seminimal mungkin (Rose et al., 2020), sesuai dengan tugas atau fungsi yang dijalankan. Dengan menerapkan prinsip ini, maka potensi penyalahgunaan hak akses dan eksposur terhadap sistem dapat diminimalisir (Kerman et al., 2020).

Privilege escalation (eskalasi hak akses) adalah sebuah proses di mana penyerang mengeksploitasi kerentanan, kesalahan desain, atau kesalahan konfigurasi pada sebuah aplikasi atau sistem operasi untuk mendapatkan akses ke sumber daya yang seharusnya tidak dapat mereka jangkau. Serangan ini memungkinkan penyerang untuk "meningkatkan" level hak akses mereka dari pengguna biasa menjadi pengguna dengan hak istimewa, seperti administrator atau root (Haimed et al., 2023).

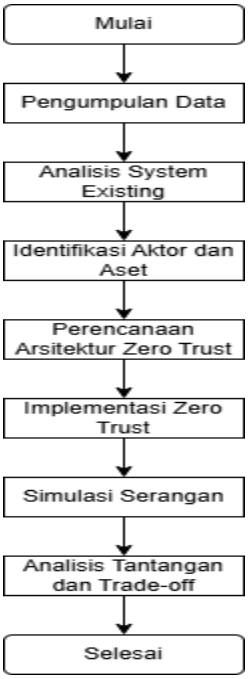
Kontrol akses adalah mekanisme yang mengatur siapa yang dapat mengakses apa dalam sistem (Penelova, 2021). Model kontrol akses yang digunakan dalam penelitian ini adalah *Role-Based Access Control* (RBAC). RBAC merupakan pendekatan manajemen akses di mana hak akses diberikan berdasarkan peran pengguna, bukan individu.

2. METODE PENELITIAN

Pendekatan ini mengacu pada standar dari NIST SP 800-207, yang menekankan prinsip *"never trust, always verify"* dengan pengambilan keputusan akses yang berbasis pada kebijakan *(policy-based access).* Sistem yang dikembangkan akan mengintegrasikan komponen utama ZTA, yaitu:

1. *Policy Engine* (PE)
2. *Policy Administrator* (PA)
3. *Policy Enforcement Point* (PEP)

2.1. Alur Penelitian



Gambar 1. Alur Penelitian

Efektivitas implementasi ZTA dalam penelitian ini diukur berdasarkan tingkat keberhasilan mitigasi (*Mitigation Success Rate*) terhadap upaya akses tidak sah. Parameter keberhasilan ditetapkan apabila sistem mampu mendeteksi dan menghentikan 100% upaya serangan pada skenario pengujian yang dirancang.

2.2. Pengumpulan Data

Pengumpulan data dilakukan untuk memperoleh informasi yang relevan dan mendukung proses perancangan, implementasi, serta evaluasi sistem keamanan berbasis *Zero Trust Architecture.* Data yang dikumpulkan terdiri dari:

2.2.1. Data Primer

Diperoleh melalui observasi langsung dan interaksi dengan sistem studi kasus. Data primer yang dikumpulkan meliputi:

1. Struktur peran pengguna (SuperAdmin, Admin, Author, Publik) dan hak akses masing-masing.
2. Mekanisme autentikasi yang digunakan pada sistem eksisting.
3. Proses pengelolaan data sensitif di dalam sistem.
4. Celah keamanan yang teridentifikasi pada autentikasi, kontrol akses, dan pencatatan aktivitas.

2.2.2. Data Sekunder

Diperoleh melalui studi pustaka dan penelusuran dokumen resmi, meliputi:

1. Standar keamanan NIST SP 800-207 sebagai acuan utama penerapan *Zero Trust Architecture.*
2. Laporan OWASP Top 10 yang memuat daftar kerentanan umum pada aplikasi *web.*

3. Jurnal ilmiah, buku, dan publikasi yang membahas konsep MFA, RBAC, prinsip *least privilege,* dan implementasi sistem keamanan berbasis kebijakan.

2.3. Analisis *Sistem Existing*

Pada tahap ini, dilakukan analisis mendalam terhadap sistem aplikasi *web* eksisting yang telah berjalan:

1. Pemetaan Struktur Pengguna dan Akses: Peneliti mengidentifikasi dan memetakan seluruh jenis pengguna (aktor) yang ada di dalam sistem,
2. Mekanisme Keamanan Awal: Peneliti melakukan analisis terhadap mekanisme keamanan yang sudah ada, terutama pada lapisan autentikasi, kontrol akses, dan pencatatan aktivitas.

2.4. Identifikasi Aktor dan Aset

Identifikasi terhadap entitas yang terlibat dalam sistem, aset yang perlu dilindungi, serta proses bisnis yang mencakup pendaftaran, pengelolaan data, dan akses *dashboard.*

Aktor. Tahap aktor ini menganalisis sistem untuk mengidentifikasi pengguna, peran, dan interaksinya dengan *platform*.

Aset. Tahap aset ini menganalisis aset sistem Halcen untuk mengidentifikasi komponen yang perlu dilindungi (*protect surface*).

2.5. Perancangan Arsitektur *Zero Trust*

Pada tahap ini, peneliti melakukan perancangan sistem keamanan berbasis *Zero Trust Architecture* (ZTA) dengan mengacu pada panduan NIST SP 800-207. Proses ini mencakup penentuan komponen utama, yaitu:

1. *Policy Engine* (PE) – dirancang sebagai pusat pengambilan keputusan berbasis kebijakan. PE memproses informasi dari identitas pengguna, status autentikasi MFA, peran *(role),* serta validasi sesi.
2. *Policy Administrator* (PA) – bertugas menyampaikan keputusan dari PE ke titik penerapan, misalnya *middleware* yang menangani autentikasi ulang atau *step-up verification.*
3. *Policy Enforcement Point* (PEP) – berada pada tingkat proteksi aplikasi (*route, gate, policy* Laravel), berfungsi sebagai pengendali langsung akses ke sumber daya.

Dalam implementasinya, rancangan arsitektur ini memperhatikan hal berikut:

1. Aktor Sistem: SuperAdmin (hak akses penuh), Admin (CRUD artikel & mitra), Author (membuat artikel & membaca *log* terbatas), Publik (pendaftaran).
2. Alur Autentikasi: semua aktor terautentikasi (SuperAdmin, Admin, Author) wajib *login* dengan kredensial, dilanjutkan verifikasi MFA. Setelah validasi berhasil, pengguna diarahkan ke *dashboard* sesuai hak akses.
3. Kebijakan Akses: setiap permintaan akses diverifikasi melalui alur: validasi identitas → MFA → *role check* → keputusan akses → pencatatan *log.*

2.6. Implementasi *Zero Trust* Ke Sistem

Implementasi arsitektur *Zero Trust* pada aplikasi *web* akan difokuskan pada pengembangan modul-modul keamanan utama sebagai berikut:

1. Perancangan Autentikasi *Multi-Faktor* (MFA) untuk memperkuat lapisan verifikasi identitas, sistem akan dirancang untuk mendukung autentikasi dua langkah melalui OTP *(One-Time Password).* Rencananya, fitur ini akan diintegrasikan dengan aplikasi *authenticator* untuk memberikan lapisan keamanan tambahan saat pengguna melakukan *login.*
2. Perancangan Kontrol Akses *(Role Base Access Control & Least Privilege)* Akan dibangun sebuah sistem kontrol akses berbasis peran, yang menerapkan prinsip hak akses minimal *(least privilege).* Dalam rancangan ini, hak akses akan dikelompokkan dan dibatasi secara tegas berdasarkan peran pengguna yang telah didefinisikan (contoh: SuperAdmin, Admin) untuk memastikan pengguna hanya bisa mengakses fitur sesuai tugasnya.
3. Perancangan Sistem *Logging*: Untuk kebutuhan pemantauan dan audit, akan dirancang mekanisme pencatatan aktivitas pengguna (*log activity*).

2.7. Simulasi Serangan

Pada tahap ini direncanakan sebuah simulasi serangan siber yang berfokus pada ancaman *lateral movement attack* melalui teknik *session hijacking.*

2.7.1. Rencana Skenario Serangan

Skenario serangan yang akan disimulasikan meliputi beberapa tahap berikut:

1. *Session Hijacking* Dengan kredensial tersebut, penyerang akan mencoba melakukan *login* untuk mengambil alih sesi pengguna yang sah.
2. *Privilege Escalation (Lateral Movement)* Setelah *login* sebagai Admin, penyerang akan mencoba mengakses fitur yang seharusnya hanya bisa digunakan oleh SuperAdmin, misalnya /program-layanan.

2.8. Analisis Tantangan dan *Trade-off*

Implementasi ZTA dalam penelitian ini menunjukkan bahwa keamanan yang lebih ketat membawa konsekuensi pada efisiensi operasional aplikasi. Tantangan utama yang ditemukan adalah adanya peningkatan beban kerja aplikasi (*overhead*) karena sistem harus melakukan verifikasi berkelanjutan pada setiap permintaan (*request*) yang masuk ke *middleware*. Selain itu, penerapan *step-up*

verification pada fitur-fitur kritis menambah hambatan bagi pengguna (*user friction*) karena kewajiban autentikasi ulang.

3. PEMBAHASAN

3.1. Pengumpulan Data

Tahap awal penelitian dilakukan dengan mengumpulkan data primer dan sekunder. Hasil dari tahapan ini mendefinisikan *protect surface* (permukaan yang harus dilindungi) dan celah keamanan yang ada.

3.1.1. Pemetaan Entitas dan Hak Akses (Data Primer)

Berdasarkan observasi langsung terhadap sistem, ditemukan empat klasifikasi pengguna dengan struktur hak akses yang masih memiliki risiko tumpang tindih:

1. SuperAdmin*:* Memegang kendali penuh (*Full Control*) terhadap seluruh konten
2. Admin: Memiliki otoritas manajemen pada *landing page,* artikel, dan mitra.
3. Author: Terfokus pada produksi konten artikel (*create, read, update*).
4. Publik: Entitas luar yang hanya berinteraksi melalui formulir pendaftaran dalam jaringan.

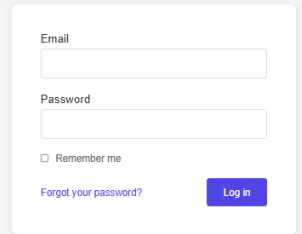
3.1.2. Identifikasi Aset Sensitif (Data Sekunder)

Data sekunder yang dikelola sistem mencakup Informasi Identitas Pribadi (PII) peserta, seperti nama, email, nomor WhatsApp, dan alamat. Mengacu pada regulasi perlindungan data (seperti GDPR), aset ini dikategorikan sebagai data sensitif yang memerlukan proteksi ketat.

3.2. Temuan Celah Keamanan pada Sistem Eksisting

Ditemukan tiga kerentanan kritis yang menjadi urgensi penerapan *Zero Trust Architecture* berdasarkan standar *OWASP Top 10 2021*:

1. Kegagalan Kontrol Akses (*Broken Access Control* - A01:2021): Belum diterapkannya prinsip *least privilege*. Hasil analisis menunjukkan bahwa pengguna dengan peran Admin secara teknis memiliki kapabilitas yang hampir setara dengan SuperAdmin, yang meningkatkan risiko eskalasi hak akses (*privilege escalation*).
2. Kegagalan Identifikasi dan Autentikasi (A07:2021): Sistem hanya mengandalkan satu faktor autentikasi (*password*). Hal ini menunjukkan ketiadaan verifikasi berlapis,
3. *Monitoring* dan *Logging* (A09:2021)**:** Sistem belum mampu mencatat aktivitas pengguna secara mendalam. Ketiadaan *audit trail* ini menyebabkan sulitnya melakukan deteksi dini terhadap perilaku anomali atau ancaman dari dalam (*insider threats*).



Gambar 2 Gambar *Login* Tanpa *Zero Trust* MFA

3.3. Identifikasi Aktor dan Aset

Setelah menemukan celah keamanan, langkah selanjutnya dalam kerangka kerja *Zero Trust* adalah menentukan apa yang harus dilindungi (*Protect Surface*).

3.3.1. Struktur Peran dan Hak Akses

Berdasarkan hasil analisis sistem, aktor diklasifikasikan bukan hanya berdasarkan jabatan, tetapi berdasarkan kebutuhan akses minimum. Penemuan di lapangan menunjukkan adanya anomali peran yang harus diperbaiki:

1. *Privileged User* (SuperAdmin)*:* Pengguna dengan otoritas tertinggi. Dalam model *Zero Trust*, akun ini merupakan risiko terbesar jika terjadi kompromi, sehingga memerlukan proteksi MFA mutlak.
2. Standard Admin & Author: Aktor yang bertanggung jawab atas konten. Analisis menunjukkan bahwa pada sistem lama, peran Author memiliki hak akses yang melampaui fungsinya (setara Admin), sehingga diperlukan restrukturisasi peran melalui RBAC.
3. *Unprivileged User* (Publik): Entitas luar yang hanya memiliki akses interaksi terbatas pada *frontend* dan pengiriman data pendaftaran.

3.3.2. Penentuan *Protect Surface* (Aset Kritis)

Implementasi *Zero Trust Architecture* difokuskan pada perlindungan aset-aset kritis yang

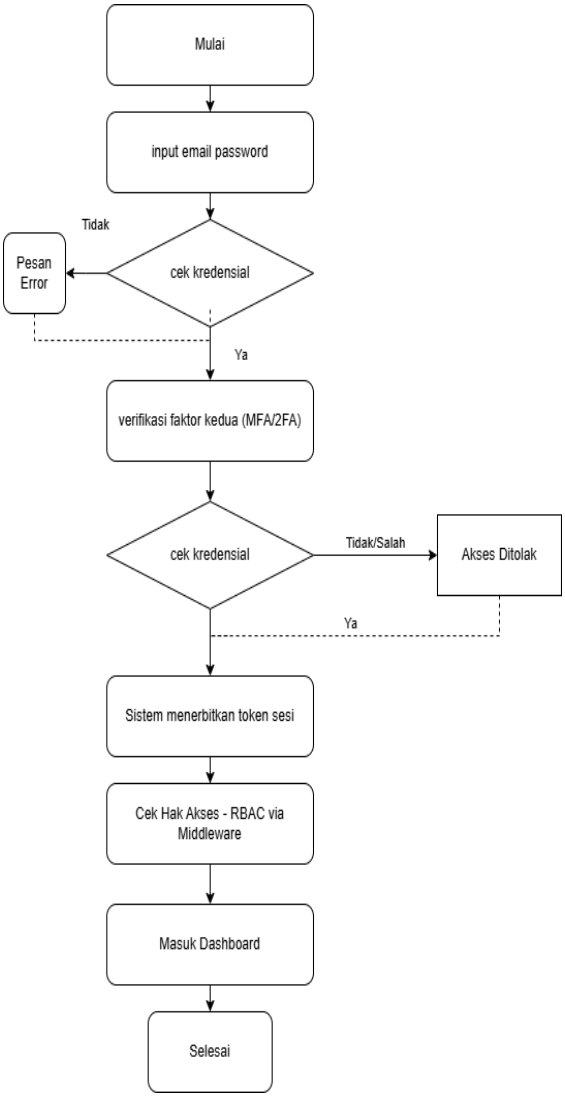
Tabel 1. Aset Kritis

No	Nama Aset	Kategori	Logika Akses *Zero Trust*
1	Data Pendaftar & Absensi	Data Pribadi (PII)	Akses dibatasi ketat hanya untuk SuperAdmin dengan verifikasi berlapis.
2	Program Layanan & *Batch*	Data Bisnis	Hanya dapat dimodifikasi oleh tingkat Admin ke atas
3	Konten *Web* (Artikel/Video)	Data Publik	Dapat dikelola oleh Author, namun modifikasi struktural memerlukan validasi peran.
4	Kredensial Pengguna	Data Identitas	Aset paling kritikal yang menjadi target utama serangan *Session Hijacking*.

3.4. Implementasi *Zero Trust*

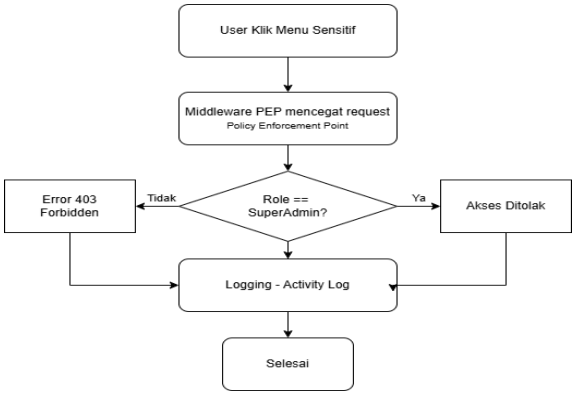
Pada tahap ini, rancangan arsitektur direalisasikan ke dalam lingkungan pengembangan *framework* Laravel. Implementasi difokuskan pada penguatan gerbang keamanan di setiap tingkatan akses.

3.4.1. Rancangan Alur Flowchart *Zero Trust* (Sistem Baru)



Gambar 4. Alur *login* dan verifikasi

Gambar 4 Alur *Login* & Verifikasi mengikuti prinsip *always verify*: setiap langkah (kredensial → MFA → *session token* → RBAC) harus lolos sebelum *user* bisa masuk *dashboard.* Percabangan "Tidak/Salah" selalu dikembalikan ke titik *input* agar tidak ada celah *bypass.*



Gambar 5. Alur akses fitur sensitif

Gambar 5 Alur Akses Fitur Sensitif menerapkan *least privilege*: *Middleware* PEP mencegat setiap *request,* lalu keputusan *Role* (*SuperAdmin* atau bukan) menentukan apakah akses diizinkan atau ditolak dengan 403 *Forbidden.* Kedua hasil (berhasil maupun gagal) tetap mengalir ke blok *Activity Log* — sehingga semua aksi terekam tanpa pengecualian.

3.4.2. Policy Engine

Policy Engine berfungsi sebagai otak yang melakukan evaluasi terhadap identitas dan peran pengguna.

1. Logika Teknis: Diimplementasikan pada *Authenticated Session Controller.* Sistem tidak hanya mengecek kecocokan *email* dan *password*, tetapi langsung melakukan validasi *role* saat percobaan *login* dilakukan.
2. Analisis: Jika pengguna mencoba masuk dengan peran yang tidak terdaftar atau tidak memiliki izin akses ke area administratif, PE akan secara otomatis memutus sesi (*force logout*) dan memberikan pesan penolakan.

```php
if ($user->hasAnyRole(['superAdmin', 'admin', 'author'])) {
    return redirect()->route('admin.dashboard');
}
```

Gambar 6. kode sumber *Policy Engine*

Jika peran tidak sah, pengguna akan dipaksa *logout*:

```php
Auth::logout();
return redirect()->route('login')->withErrors([
    'email' => 'Akun Anda tidak memiliki hak akses.',
]);
```

Gambar 7. Kode fungsi *Policy Engine*

3.4.3. Policy Administrator

Bertugas mengelola alur keputusan setelah PE memberikan verifikasi awal.

1. Logika Teknis: Diimplementasikan melalui *Middleware* khusus, seperti *EnsureTwoFactorCodeIsVerified.*
2. Analisis: PA berfungsi sebagai lapisan verifikasi transisi. Meskipun *login* awal berhasil, PA akan

menahan pengguna di "zona karantina" (halaman *input* OTP) hingga faktor kedua (MFA) terpenuhi. Ini menjamin bahwa akses ke sumber daya sistem hanya dibuka setelah seluruh kebijakan keamanan terpenuhi secara kumulatif.

```
if (auth()->check() && auth()->user()->google2fa_secret) {

    if (!session('2fa_verified')) {
        return redirect()->route('2fa.form')->with('error', 'Silakan verifikasi kode 2FA terlebih dahulu.');
    }
}
```

Gambar 8. Kode Sumber *Policy Administrator*

3.4.4. Policy Enforcement Point

PEP adalah garda terdepan yang menjaga setiap aset (*Protect Surface*).

1. **Logika Teknis:** Menggunakan sistem *Route Guard* di Laravel yang menggabungkan beberapa *middleware* sekaligus *(auth, verified, 2fa, dan role).*
2. **Analisis:** Dengan menerapkan PEP pada setiap *route*, sistem menerapkan prinsip ***"Verify Explicitly"***. Setiap *request* yang masuk wajib membawa *token* validasi yang lengkap. Tanpa atribut yang sesuai (misal: *user* adalah 'Author' tapi mencoba akses route 'SuperAdmin'), PEP akan langsung memblokir akses dan mengembalikan respon *403 Forbidden*.

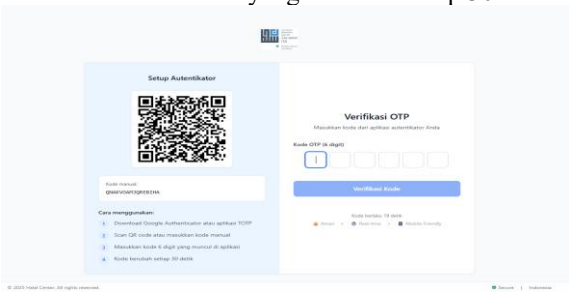
```
if ($user->hasAnyRole(['superAdmin', 'admin', 'author'])) {
    return redirect()->route('admin.dashboard');
}
```

Gambar 9. Kode Sumber *Policy Enforcement Point*

3.4.5. Multi Factor Authentication

MFA menjadi instrumen utama dalam mengatasi celah *Identification and Authentication Failures*.

1. Mekanisme**:** Menggunakan algoritma TOTP (*Time-based One-Time Password*) yang terhubung dengan aplikasi Google Authenticator.
2. Analisis**:** Penambahan MFA secara signifikan meningkatkan ambang batas keamanan. Bahkan jika data primer (kata sandi) bocor, aset tetap aman karena penyerang memerlukan akses fisik ke perangkat pengguna untuk mendapatkan kode verifikasi dinamis yang berubah setiap 30 detik.



Gambar 10. Tampilan MFA

3.4.6. Least Privilege and Role Base Access Control

Pembatasan hak akses diterapkan hingga ke tingkat antarmuka pengguna (*UI-level*) dan basis data.

1. **Mekanisme:** Menggunakan direktif @hasanyrole pada *blade template* Laravel.
2. **Analisis:** Dengan membatasi tampilan menu berdasarkan peran, sistem meminimalkan *attack surface*. Admin hanya melihat apa yang perlu mereka kelola, sementara akses ke data sensitif pendaftaran benar-benar diisolasi hanya untuk SuperAdmin.

```
@role('superAdmin')
        <p class="uppercase text-xs text-gray-600 mb-4 mt-4 tracking-wider">Program Layanan & Peserta</p>

        <a href="{{ route('admin.program-layanan.index') }}"
           class="mb-3 capitalize font-medium text-sm hover:text-teal-600 transition ease-in-out duration-500">
            <i class="fad fa-cogs text-blue-500 text-xs mr-2"></i>
            Program Layanan
        </a>
```

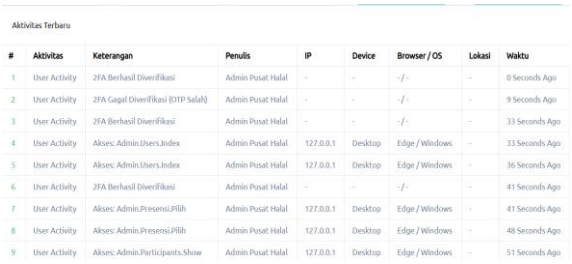
Gambar 11. Kode sumber *Role Base Access Control*

3.4.7. Log Activity Monitoring

Fitur *log* aktivitas diimplementasikan menggunakan *package spatie/laravel-activitylog*, dan hasil aktivitas ditampilkan langsung di *dashboard* SuperAdmin dalam tabel Aktivitas Terbaru.

Tabel ini menampilkan:

1. Aktivitas yang dilakukan
2. Deskripsi aktivitas
3. Nama pelaku
4. Waktu aktivitas (menggunakan *diffForHumans()* dari *Carbon*).

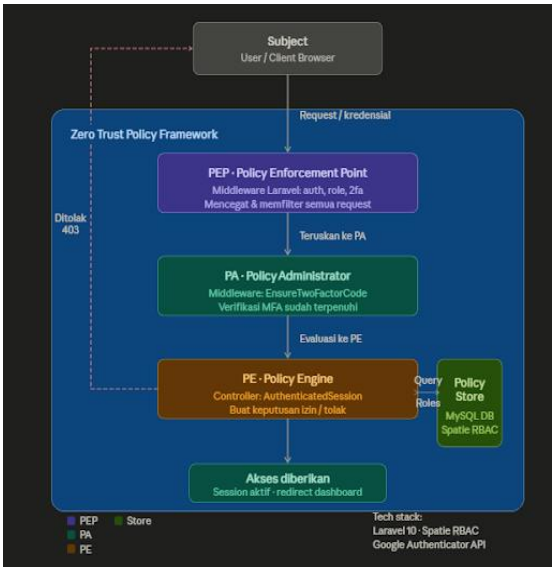


Gambar 13. *Log Activity*

3.5. Simulasi Serangan dan Pengujian Keamanan

Untuk membuktikan efektivitas arsitektur *Zero Trust* yang telah diimplementasikan, dilakukan simulasi serangan menggunakan skenario ancaman dari dalam (*insider threat*) dan pencurian identitas.

3.5.1. Diagram Zero Trust Architecture



Gambar 14 Diagram *Zero Trust Architecture*

1. ***Subject to PEP (The Request):*** Subjek *(User)* mengirimkan permintaan akses. Permintaan ini pertama kali dicegat oleh ***Policy Enforcement Point*** **(PEP)** yang direpresentasikan oleh lapisan *Middleware* Laravel *(auth, verified, 2fa,* dan *role).*
2. **PEP *to* PA *(Verification Flow)*:** PEP meneruskan status autentikasi ke ***Policy Administrator*** **(PA)** melalui *middleware EnsureTwoFactorCodeIsVerified.* PA bertugas memastikan bahwa alur verifikasi (seperti *input* kode MFA dari *Google Authenticator*).
3. **PA *to* PE *(The Decision): Policy Engine* (PE)** yang terletak pada *AuthenticatedSessionController* menerima data identitas dan MFA *token.* PE melakukan evaluasi mendalam terhadap kredensial dan status akun.
4. **PE *to Policy Store (Data Validation):*** PE melakukan *query* ke ***Policy Store*** (*Database* MySQL yang dikelola oleh *Spatie Role Package*) untuk memvalidasi apakah peran pengguna memiliki otoritas terhadap sumber daya (*resource*) yang diminta.
5. ***Decision Flow (Allow/Deny):*** Berdasarkan validasi tersebut, PE memberikan instruksi *Allow* (lanjut ke *Dashboard*) atau *Deny* (alihkan ke *login/error* 403). Seluruh proses ini dicatat dalam *Activity Log* sebagai bagian dari fungsi *monitoring*.

3.6. Simulasi Eskalasi Hak Akses

Skenario ini menguji ketangguhan *Policy Enforcement Point* (PEP) dalam menjaga batas-batas hak akses antar peran (*role*).

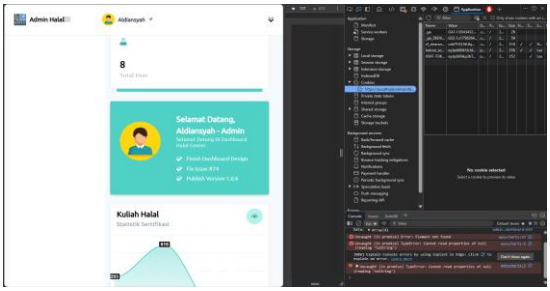
1. **Skenario:** Pengguna dengan peran 'Admin' mencoba mengakses secara paksa (*forced browsing*) ke URL /admin/program-layanan, yang merupakan rute khusus untuk 'SuperAdmin' guna mengelola data pendaftar sensitif.
2. **Hasil Pengujian:** Sistem segera mendeteksi bahwa atribut *role* pada sesi pengguna tidak memenuhi syarat yang ditetapkan oleh *Middleware* PEP. Permintaan akses diblokir seketika.
3. **Analisis:** Kegagalan akses ini (Gambar 15) membuktikan penerapan prinsip ***Least Privilege***. Tidak adanya kepercayaan implisit berarti meskipun pengguna memiliki akun sah di sistem, ia tidak otomatis memiliki akses ke seluruh bagian aplikasi. Setiap *request* divalidasi secara eksplisit berdasarkan kebijakan akses yang ketat.

Gambar 3 Error Hak Akses

3.7. Simulasi *Lateral Movement*

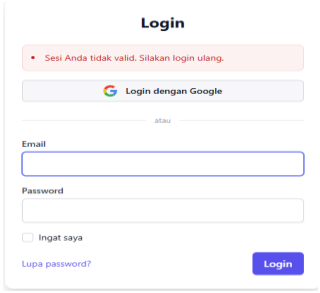
Skenario kedua menguji filosofi *"Never Trust"* terhadap sesi aktif yang dicuri.

1. Skenario: Penyerang berperan sebagai *insider* yang berhasil mencuri *cookie* sesi (halcen_session dan XSRF-TOKEN) milik SuperAdmin. Penyerang mencoba menyuntikkan (*inject*) *cookie* tersebut ke peramban lain untuk melewati proses *login* (Gambar 16).
2. Hasil Pengujian: Sistem mendeteksi adanya upaya akses dari konteks yang mencurigakan. Karena *Policy Administrator* (PA) mewajibkan verifikasi berkelanjutan, penyerang tidak langsung mendapatkan akses penuh. Sistem memaksa pemutusan sesi (*forced logout*) atau tetap meminta verifikasi MFA (Gambar 17).
3. Analisis: Keberhasilan pencegahan ini menunjukkan bahwa dalam *Zero Trust Architecture*, sebuah "sesi" tidak lagi dianggap sebagai bukti identitas yang permanen.



Gambar 4 Simulasi *Session Hijacking*

Ketika admin ingin mencuri *session cookie* agar dia bisa masuk sebagai *SuperAdmin* nantinya, maka akan dicegah dan akan di keluarkan otomatis.



Gambar 5 *Session* berhasil di cegah

3.7.1. Pengukuran Latensi Autentikasi *(Response Time)*

Pengujian performa dilakukan untuk mengukur *Task Completion Time* (Total Waktu Penyelesaian Login) yang merepresentasikan pengalaman pengguna (*User Experience),* dimulai dari pengguna memasukkan kredensial tahap pertama hingga proses memasukkan kode *Multi-Factor Authentication* (MFA).

Tabel 2. Hasil Uji Coba Latensi *Login* (10 Replikasi)

Percobaan Ke-	Sistem Eksisting (Detik)	Sistem Zero Trust / MFA (Detik)
1	3,2	10,5
2	3,5	11,2
3	4,1	9,8
4	3,8	12,0
5	3,4	10,2
6	3,6	13,1
7	4,0	11,5
8	3,3	10,8
9	3,9	12,5
10	3,5	11,0
Rata-rata	3,63 Detik	11,26 Detik

Berdasarkan Tabel 2, penerapan *Zero Trust Architecture* dengan MFA meningkatkan total waktu login rata-rata dari 3,63 detik menjadi 11,26 detik. Peningkatan waktu sekitar 7-8 detik ini adalah hal yang logis dan wajar, mengingat pengguna membutuhkan jeda waktu fisik untuk menerima dan menginput kode keamanan dari perangkat tambahan mereka (seperti email atau *authenticator app*).

3.8. Matriks Pengujian Mitigasi Serangan (8 *Test Cases)*

Pengujian dilakukan terhadap 8 skenario serangan untuk menghitung *Mitigation Success Rate*.

Tabel 3. Matriks Hasil Pengujian Keamanan Terstruktur

No	Jenis Serangan	Skenario Pengujian	Expected Result	Actual Result	Status
1	Kebocoran Kredensial	Login hanya dengan Email & Password	Akses Tertahan (Minta OTP)	Akun tidak bisa masuk	Lulus
2	Penelusuran Paksa	Admin akses URL /admin/user	Respon 403 Forbidden	Akses Ditolak	Lulus
3	Eskalasi Hak Akses	Author akses fitur Program	Respon 403 Forbidden	Akses Ditolak	Lulus
4	Pembajakan Sesi	Injeksi Cookie ke Browser lain	Sesi otomatis dihentikan	Logout Paksa	Lulus
5	Pergerakan Lateral	Penyerang pindah antar menu SA	Verifikasi Role berkelanjutan	Akses Terputus	Lulus
6	Autentikasi rusak	Bypass halaman MFA via URL	Redireksi otomatis ke MFA	Kembali ke MFA	Lulus
7	Pengulangan token	Pakai token MFA yang sudah expired	Token ditolak	Akses Ditolak	Lulus
8	Akses tanpa autentikasi	Akses dashboard tanpa login	Redireksi ke /login	Berhasil Dicegat	Lulus

3.9. Perbandingan Sebelum vs Sesudah ZTA

Berdasarkan hasil pengujian dan observasi yang telah dilakukan, perbedaan signifikan antara mekanisme keamanan sebelum dan sesudah penerapan *Zero Trust* dirangkum dalam Tabel 4 di bawah ini.

Tabel 4. Perbandingan Sebelum vs Sesudah ZTA

Parameter Perbandingan	Sistem Eksisting (Sebelum ZTA)	Sistem Baru (Sesudah ZTA)
Metode Autentikasi	Faktor tunggal (*Single Factor*) hanya *email* & *password*.	Verifikasi berlapis dengan *Multi-Factor Authentication* (MFA) berbasis TOTP .
Prinsip Hak Akses	Risiko tumpang tindih; peran Admin hampir setara SuperAdmin.	Penerapan *Least Privilege* melalui RBAC yang ketat.
Keamanan Sesi	*Session cookie* bisa disalahgunakan jika bocor.	Verifikasi berkelanjutan; sesi otomatis dihentikan jika terdeteksi anomali/pencurian *cookie* .
Monitoring	Belum mampu mencatat aktivitas pengguna secara mendalam.	*Logging* aktivitas secara *real-time* (siapa, kapan, melakukan apa) .
Kecepatan Login	Rata-rata 3,63 detik.	Rata-rata 11,26 detik (karena proses MFA).

4. KESIMPULAN DAN SARAN

Berdasarkan hasil implementasi dan evaluasi yang telah dilakukan, penelitian ini memberikan jawaban atas permasalahan yang telah diidentifikasi pada Bab I. Tiga kelemahan utama pada sistem

eksisting berhasil diatasi melalui penerapan prinsip *Zero Trust Architecture* (ZTA), dengan rincian sebagai berikut:

1. Mekanisme autentikasi satu faktor yang rentan serangan. Permasalahan ini terjawab melalui penerapan *Multi-Factor Authentication* (MFA) menggunakan OTP berbasis aplikasi Authenticator. Dengan adanya lapisan verifikasi tambahan, risiko serangan seperti *phishing, credential stuffing,* maupun pencurian akun dapat diminimalisasi.
2. Pengaturan hak akses berhasil diperketat melalui *Role-Based Access Control* (RBAC). Dengan mekanisme ini, setiap pengguna hanya memperoleh hak akses sesuai perannya (SuperAdmin, Admin, Author, Publik). Hal ini mencegah terjadinya eskalasi hak akses dan penyalahgunaan kewenangan di dalam sistem dan sudah sesuai dengan prinsip *zero trust* bagian *least privilege.*
3. Pencatatan aktivitas pengguna yang belum mendukung audit keamanan
   Permasalahan ini diatasi dengan penerapan fitur *logging* aktivitas. Seluruh aktivitas penting, seperti *login, logout,* dan perubahan data, berhasil dicatat sehingga dapat mendukung proses audit, evaluasi keamanan, dan deteksi dini terhadap aktivitas mencurigakan.

Dengan demikian, penerapan *Zero Trust Architecture* pada sistem aplikasi Program Halal terbukti mampu meningkatkan keamanan secara signifikan melalui penguatan autentikasi, pembatasan hak akses, dan pencatatan aktivitas yang komprehensif.

4.1. Saran dan Pengembangan

Meskipun penelitian ini berhasil menerapkan prinsip utama ZTA, masih terdapat ruang untuk pengembangan lebih lanjut. Beberapa rekomendasi yang dapat dijadikan acuan, antara lain:

1. Pengembangan *Trust Scoring* Dinamis
   Menambahkan algoritma penilaian tingkat kepercayaan pengguna dengan mempertimbangkan faktor seperti alamat IP, *device ID*, lokasi geografis, waktu akses, dan pola *login.* Hal ini dapat menggantikan autentikasi statis menjadi lebih adaptif dan kontekstual.
2. Pemantauan Berbasis *Event Log*
   Menambahkan notifikasi otomatis kepada SuperAdmin apabila terdeteksi aktivitas mencurigakan, seperti percobaan *login* gagal berulang, akses dari perangkat baru, atau perubahan data sensitif.
3. Skalabilitas Implementasi ZTA
   Melakukan pengembangan lebih lanjut agar sistem mampu mengimplementasikan komponen ZTA secara menyeluruh sesuai kerangka kerja NIST SP 800-207, termasuk mekanisme evaluasi sinyal kontekstual dan orkestrasi kebijakan keamanan lintas aplikasi.

DAFTAR PUSTAKA

Abdulazeez, A. M., Salim, B. W., Zeebaree, D. Q., & Doghramachi, D. (2020). Comparison of VPN Protocols at Network Layer Focusing on Wire Guard Protocol. International Journal of Interactive Mobile Technologies, 14(18). https://doi.org/10.3991/ijim.v14i18.16507

Alwi, E. I., Herdianti, H., & Umar, F. (2020). Analisis Keamanan Website Menggunakan Teknik Footprinting dan Vulnerability Scanning. INFORMAL: Informatics Journal, 5(2). https://doi.org/10.19184/isj.v5i2.18941

Aslan, Ö., Aktuğ, S. S., Ozkan-Okay, M., Yilmaz, A. A., & Akin, E. (2023). A Comprehensive Review of Cyber Security Vulnerabilities, Threats, Attacks, and Solutions. In Electronics (Switzerland) (Vol. 12, Number 6). MDPI. https://doi.org/10.3390/electronics12061333

Azad, M. A., Abdullah, S., Arshad, J., Lallie, H., & Ahmed, Y. H. (2024). Verify and trust: A multidimensional survey of zero-trust security in the age of IoT. Internet of Things, 27, 101227. https://doi.org/https://doi.org/10.1016/j.iot.2024.101227

Buck, C., Olenberger, C., Schweizer, A., Völter, F., & Eymann, T. (2021). Never trust, always verify: A multivocal literature review on current knowledge and research gaps of zero-trust. Computers and Security, 110. https://doi.org/10.1016/j.cose.2021.102436

Campbell, M. (2020). Beyond Zero Trust: Trust Is a Vulnerability. Computer, 53(10), 110–113. https://doi.org/10.1109/MC.2020.3011081

Edo, O. C., Tenebe, T., Etu, E., Ayuwu, A., Emakhu, J., & Adebiyi, S. (2022). Zero Trust Architecture: Trend and Impacton Information Security. International Journal of Emerging Technology and Advanced Engineering, 12(7). https://doi.org/10.46338/ijetae0722_15

Elanda, A., & Buana, R. L. (2020). Analisis Keamanan Sistem Informasi Berbasis Website Dengan Metode Open Web Application Security Project (OWASP) Versi 4: Systematic Review. CESS (Journal of Computer Engineering, System and Science), 5(2). https://doi.org/10.24114/cess.v5i2.17149

Fronita, M. (2023). ANALISIS CELAH KEAMANAN WEBSITE SITASI MENGGUNAKAN VULNERABILITY ASSESSMENT. Jurnal Ilmiah Rekayasa Dan Manajemen Sistem Informasi, 9(1). https://doi.org/10.24014/rmsi.v9i1.21823

Ge, Y., & Zhu, Q. (2024). GAZETA: GAme-Theoretic ZEro-Trust Authentication for Defense Against

Lateral Movement in 5G IoT Networks. IEEE Transactions on Information Forensics and Security, 19. https://doi.org/10.1109/TIFS.2023.3326975

Glemser, T. (2022). OWASP Top 10. Datenschutz Und Datensicherheit - DuD, 46(11). https://doi.org/10.1007/s11623-022-1685-5

Haimed, I. B., Albahar, M., & Alzubaidi, A. (2023). Exploiting Misconfiguration Vulnerabilities in Microsoft's Azure Active Directory for Privilege Escalation Attacks. Future Internet, 15(7). https://doi.org/10.3390/fi15070226

Herrada-Lores, S., Iniesta-Bonillo, M. Á., & Estrella-Ramón, A. (2022). Weaknesses and strengths of online marketing websites. Spanish Journal of Marketing - ESIC, 26(2). https://doi.org/10.1108/SJME-11-2021-0219

Hu, B., Tang, W., & Xie, Q. (2022). A two-factor security authentication scheme for wireless sensor networks in IoT environments. Neurocomputing, 500, 741–749. https://doi.org/https://doi.org/10.1016/j.neucom.2022.05.099

Kerman, A., Borchert, O., Rose, S., Division, E., & Tan, A. (2020). Implementing a Zero Trust Architecture. NIST Computer Security Resource Center, (July).

Kumari, P., & Jain, A. K. (2023). A comprehensive study of DDoS attacks over IoT network and their countermeasures. In Computers and Security (Vol. 127). https://doi.org/10.1016/j.cose.2023.103096

Kumi, S., Lim, C. H., Lee, S. G., Oktian, Y. O., & Witanto, E. N. (2021). Automatic Detection of Security Misconfigurations in Web Applications. Lecture Notes in Networks and Systems, 149. https://doi.org/10.1007/978-981-15-7990-5_8

Laverdière, M. A., Julien, K., & Merlo, E. (2021). RBAC protection-impacting changes identification: A case study of the security evolution of two PHP applications. Information and Software Technology, 139. https://doi.org/10.1016/j.infsof.2021.106630

Li, J. (2020). Vulnerabilities mapping based on OWASP-SANS: A survey for static application security testing (SAST). Annals of Emerging Technologies in Computing, 4(3). https://doi.org/10.33166/AETiC.2020.03.001

Li, Y., & Liu, Q. (2021). A comprehensive review study of cyber-attacks and cyber security; Emerging trends and recent developments. Energy Reports, 7. https://doi.org/10.1016/j.egyr.2021.08.126

Mary, T., & Febriyani, N. (2025). Peningkatan Keamanan Sistem Informasi Berbasis Laravel 12 dengan Rate Limiting dan Role-Based Access Control (RBAC). Jurnal Teknologi Dan Sistem Informasi Bisnis, 7(3), 473–481. https://doi.org/10.47233/jteksis.v7i3.1976

Mr Pradeep Nayak, Mohammed Sufiyan, Monisha N S, Moolly Gautami Bhaskar, & Mohan Raju. (2022). Review Paper on Cyber Security and Types of Cyber Attacks. International Journal of Advanced Research in Science, Communication and Technology, 732–735. https://doi.org/10.48175/IJARSCT-7043

Pakpahan, A. V. (2024). Convolutional Neural Network and Haversine Formula in Presence System for Easy Attendance. 2024 Asian Conference on Communication and Networks (ASIANComNet), 1–6. https://doi.org/10.1109/ASIANComNet63184.2024.10810512

Penelova, M. (2021). Access Control Models. Cybernetics and Information Technologies, 21(4). https://doi.org/10.2478/cait-2021-0044

Phiayura, P., & Teerakanok, S. (2023). A Comprehensive Framework for Migrating to Zero Trust Architecture. IEEE Access, 11. https://doi.org/10.1109/ACCESS.2023.3248622

Prayama, D., Yuhefizar, & Amelia Yolanda. (2021). Protokol HTTPS, Apakah Benar-benar Aman? Journal of Applied Computer Science and Technology, 2(1). https://doi.org/10.52158/jacost.v2i1.118

Priyawati, D., Rokhmah, S., & Utomo, I. C. (2022). Website Vulnerability Testing and Analysis of Internet Management Information System Using OWASP. International Journal of Computer and Information System (IJCIS) Peer Reviewed-International Journal, 3(3). https://doi.org/10.29040/ijcis.v3i3.90

Putra, F. P. E., Efendi, R. W., Tamam, A. B., & Pramadi, W. A. (2025). Tren dan Praktik Terbaik dalam Pengembangan Web Berbasis API : Kajian Literatur terhadap Framework Laravel dan React. Infomatek, 27(1), 165–178. https://doi.org/10.23969/infomatek.v27i1.25122

Rafi, M., Ihsan, I., & Voutama, A. (2025). Penerapan Metode NIST Dalam Analisis Forensik Digital Pasca Serangan Siber (Studi Kasus : Pt.Analis Digital Forensik) 1 (Vol. 8, Number 1).

Ray, P. P. (2023). Web3: A comprehensive review on background, technologies, applications, zero-trust architectures, challenges and future directions. In Internet of Things and Cyber-Physical Systems (Vol. 3). https://doi.org/10.1016/j.iotcps.2023.05.003

Rose, S., Borchert, O., Mitchell, S., & Connelly, S. (2020). Zero Trust Architecture. https://doi.org/10.6028/NIST.SP.800-207

Sadeghi, S., & Hadavi, M. A. (2021). Automatic Black-Box Detection of Resistance Against CSRF

Vulnerabilities in Web Applications. Journal of Computing and Security Research Article, 8(1).

Sarkar, S., Choudhary, G., Shandilya, S. K., Hussain, A., & Kim, H. (2022). Security of Zero Trust Networks in Cloud Computing: A Comparative Review. In Sustainability (Switzerland) (Vol. 14, Number 18). MDPI. https://doi.org/10.3390/su141811213

Shore, M., Zeadally, S., & Keshariya, A. (2021). Zero Trust: The What, How, Why, and When. In Computer (Vol. 54, Number 11, pp. 26–35). IEEE Computer Society. https://doi.org/10.1109/MC.2021.3090018

Suleski, T., Ahmed, M., Yang, W., & Wang, E. (2023). A review of multi-factor authentication in the Internet of Healthcare Things. In Digital Health (Vol. 9). SAGE Publications Inc. https://doi.org/10.1177/20552076231177144

Teerakanok, S., Uehara, T., & Inomata, A. (2021). Migrating to Zero Trust Architecture: Reviews and Challenges. In Security and Communication Networks (Vol. 2021). Hindawi Limited. https://doi.org/10.1155/2021/9947347

Villegas-Ch, W., & García-Ortiz, J. (2023). Authentication, access, and monitoring system for critical areas with the use of artificial intelligence integrated into perimeter security in a data center. Frontiers in Big Data, 6. https://doi.org/10.3389/fdata.2023.1200390

Zaky, K., & Saxe, D. H. (2022). Multi-factor Authentication. IDPro Body of Knowledge, 1(10). https://doi.org/10.55621/idpro.92