CONTEN : Computer and Network Technology
Vol. 5, No. 2, Desember 2025, hlm. 48-56

e-ISSN: 2797-5274
p-ISSN: 2809-9788

PENERAPAN JSON API UNTUK VERIFIKASI MINIMUM BASELINE SECURITY STANDARD (MBSS) PADA PERANGKAT CISCO NEXUS

Dadang Iskandar Mulyana[1] Arief Sofyan*[2]

[1,2] Teknik Informatika, Sekolah Tinggi Ilmu Komputer Cipta Karya Informatika (STIKOM CKI), Jl. Raden Inten II, Duren Sawit, Jakarta Timur, Jakarta, Indonesia

Email: [1] mahvin2012@gmail.com, [2] sfynarief18@gmail.com
*Penulis Korespondensi

Abstrak

Verifikasi *Minimum Baseline Security Standard* (MBSS) merupakan prosedur penting untuk memastikan bahwa konfigurasi keamanan pada perangkat jaringan telah sesuai sebelum dioperasikan dalam jaringan produksi. Dalam lingkungan *Cisco Nexus ACI* yang berskala besar dan kompleks, proses verifikasi memerlukan pendekatan yang efisien, akurat, dan terdokumentasi. Namun, pendekatan tradisional yang digunakan selama ini sering kali kurang praktis karena data konfigurasi tersebar, tidak terstruktur, dan membutuhkan banyak langkah untuk diakses. Penelitian ini bertujuan untuk menerapkan metode GET API dalam menampilkan data verifikasi MBSS perangkat *Cisco Nexus ACI* secara langsung dan ringkas dalam format JavaScript Object Notation (JSON). Penelitian dilakukan dengan studi literatur dan eksplorasi API menggunakan client Postman terhadap beberapa endpoint konfigurasi pada *Cisco ACI*. Hasil yang diperoleh menunjukkan bahwa metode GET API mampu menyajikan data konfigurasi MBSS secara spesifik dalam satu langkah aksi per poin dan per perangkat. Dengan format JSON yang terstruktur, proses verifikasi menjadi lebih mudah didokumentasikan dan mencapai tingkat keberhasilan & relevansi penarikan data konfigurasi mencapai 100% pada seluruh 6 parameter MBSS yang diuji. Pendekatan ini dapat menjadi alternatif yang relevan untuk meningkatkan efisiensi verifikasi keamanan jaringan di infrastruktur skala besar.

Kata Kunci: *MBSS, Cisco Nexus ACI, API, JSON, verifikasi keamanan konfigurasi jaringan*

Abstract

Minimum Baseline Security Standard (MBSS) verification is a critical procedure to ensure that network device security configurations meet predefined requirements before integration into the production environment. In large-scale and complex Cisco Nexus ACI environments, the verification process demands an efficient, accurate, and well-documented approach. However, traditional methods commonly used are often impractical due to scattered configuration data, lack of structure, and multi-step retrieval processes. This research aims to explore the use of the GET API method to display MBSS verification data of Cisco Nexus ACI devices directly and concisely in JavaScript Object Notation (JSON) format. The study involves literature review and API exploration using the Postman client across several configuration endpoints available in Cisco ACI. The results show that the GET API method can present MBSS configuration data specifically in a single action per point and per device. With a structured JSON format, the verification process becomes easier to document and have a 100% success rate for retrieving configuration data across all 6 tested MBSS parameters. This approach presents a relevant alternative to enhance the effeciency of security verification in large-scale network infrastructures.

Keyword: *MBSS, Cisco Nexus ACI, GET API, JSON, network configuration security verification*

e-ISSN: 2797-5274
p-ISSN: 2809-9788

1. PENDAHULUAN

Keamanan jaringan merupakan aspek mendasar dalam pengelolaan infrastruktur teknologi informasi perusahaan. Sebelum sebuah perangkat digunakan secara aktif dalam sistem jaringan, perlu dipastikan bahwa konfigurasi keamanannya telah sesuai dengan standar yang ditetapkan [1,2]. Salah satu langkah penting dalam proses ini adalah verifikasi *Minimum Baseline Security Standard* (MBSS), yaitu standar keamanan minimum yang wajib dipenuhi oleh perangkat sebelum bergabung ke jaringan produksi. *Security baseline* adalah jaminan keamanan minimum bagi sistem informasi, sekaligus menjadi persyaratan keamanan paling dasar[3]. Penerapan *Minimum Baseline Security Standard* pada perusahaan dapat meningkatkan tingkat kinerja keamanan pengoperasian dan pemeliharaan informasi, serta efisiensi manajemen, sehingga dapat memastikan sistem informasi bisnis beroperasi secara aman dan stabil[4].

Minimum Baseline Security Standard (MBSS) merupakan suatu kerangka kebijakan organisasi untuk menetapkan konfigurasi keamanan minimum yang harus dipenuhi sebelum perangkat atau sistem dioperasikan dalam lingkungan produksi[5z6]. Meskipun istilah "MBSS" sendiri belum muncul sebagai istilah resmi dalam standar internasional seperti ISO/IEC 27001, prinsip-prinsip dasarnya sangat konsisten dengan konsep *baseline configuration* yang dijelaskan dalam dokumen NIST SP 800-53 (kontrol CM-2), di mana organisasi diwajibkan untuk mengembangkan dan memelihara konfigurasi dasar sistem secara terdokumentasi dan teruji[7]. Selain itu, NIST SP 800-171 juga menegaskan pentingnya menjaga *baseline* konfigurasi sebagai bagian dari pengelolaan keamanan sistem informasi[8]. Dengan mengacu pada pedoman tersebut, MBSS menjadi landasan penting untuk memastikan bahwa perangkat telah memenuhi elemen-elemen keamanan dasar seperti autentikasi, kontrol akses, dan pembaruan sistem sebelum digunakan secara operasional[7].

Dalam konteks jaringan modern seperti *Cisco Nexus* dengan arsitektur *Application Centric Infrastructure* (ACI), proses verifikasi MBSS menjadi tantangan tersendiri[9,10]. Arsitektur ini dapat mencakup ratusan perangkat seperti 500 *switch leaf*, 50 *switch spine*, dan 7 APIC *controller* yang saling terhubung[11]. Ketika verifikasi dilakukan secara manual, waktu dan upaya yang dibutuhkan menjadi sangat besar, terutama karena antarmuka tradisional seperti CLI dan GUI APIC tidak secara langsung menyediakan informasi konfigurasi dalam format yang sederhana dan ringkas[12].

Sementara itu, *Cisco ACI* sebenarnya menyediakan antarmuka modern berbasis *Application Programming Interface* (API)[13]. Salah satu metode yang tersedia adalah GET API, yang memungkinkan pengambilan data konfigurasi dalam format *JavaScript Object Notation* (JSON)[13]. Format ini tidak hanya lebih ringan dan terstruktur, tetapi juga memungkinkan penyajian data secara langsung dan spesifik melalui satu langkah aksi[14,15]. API ini membuka peluang untuk menyederhanakan proses verifikasi MBSS dengan cara yang lebih praktis dan cepat[16]. Seperti disampaikan dalam penelitian sebelumnya, *Cisco ACI* mendukung definisi kebijakan jaringan secara otomatis melalui API yang mengurangi waktu konfigurasi dan risiko kesalahan[17].

Berbagai penelitian terdahulu telah membahas topologi dan arsitektur jaringan pusat data dalam upaya meningkatkan efisiensi dan skalabilitas[11,18]. Salah satu contohnya adalah kajian oleh Lebiednik dan rekan-rekannya yang membahas sejumlah desain jaringan seperti *Fat Tree, DCell*, dan *Facebook Fabric*, serta tantangan teknis seperti *latency* dan *throughput* pada skala besar[18]. Ijari juga melakukan perbandingan antara *Cisco ACI* dan *VMware NSX* dalam konteks *Software Defined Networking* (SDN), dan menyoroti keunggulan *Cisco ACI* dalam dukungan API berbasis JSON[19]. Namun, hingga saat ini belum ditemukan kajian yang secara spesifik menyoroti penerapan metode GET API dalam proses verifikasi MBSS pada perangkat *Cisco Nexus ACI*, khususnya yang berfokus pada penyajian data secara langsung dan ringkas dalam format JSON. Maka dari itu, penelitian ini diharapkan dapat mengisi celah tersebut.

Berdasarkan hal tersebut, penelitian ini bertujuan untuk mengeksplorasi bagaimana metode GET API dapat diterapkan untuk menampilkan data konfigurasi MBSS dalam format JSON secara langsung dan efisien pada perangkat *Cisco Nexus ACI*[20]. Fokus utama penelitian ini adalah memahami proses penerapan antarmuka API tersebut dalam konteks kebutuhan verifikasi keamanan jaringan yang semakin kompleks dan dinamis.

2. METODE PENELITIAN

Data yang digunakan dalam penelitian ini berupa konfigurasi sistem jaringan yang bersifat non-numerik, mencerminkan kondisi aktual parameter keamanan pada infrastruktur *Cisco Nexus ACI*[18]. Data tersebut mencakup informasi seperti daftar akun pengguna, kebijakan kata sandi, serta batasan jumlah percobaan login yang gagal[5]. Pengambilan data dilakukan melalui metode GET API, yaitu permintaan berbasis protokol HTTP untuk menarik informasi dari sistem *Cisco ACI* melalui *controller APIC*[19]. Hasil permintaan dikembalikan dalam format *JavaScript Object Notation* (JSON), yang dipilih karena sifatnya ringan, terstruktur hierarkis, dan mudah dibaca[14]. Struktur key-value pada JSON memungkinkan pencarian langsung terhadap nilai konfigurasi tertentu, misalnya panjang minimal karakter kata sandi atau

e-ISSN: 2797-5274
p-ISSN: 2809-9788

jumlah riwayat sandi yang disimpan[15]. Selain itu, format ini mendukung proses dokumentasi hasil verifikasi dan konversi data ke format lain seperti tabel atau teks[16].

Sumber data penelitian berasal dari *Cisco Application Policy Infrastructure Controller* (APIC) yang diakses melalui *Cisco APIC Sandbox*[20]. *Cisco APIC* Sandbox merupakan laboratorium publik daring resmi milik Cisco yang menyediakan simulasi infrastruktur *Cisco ACI* untuk keperluan pengujian dan pembelajaran tanpa memerlukan koneksi VPN perusahaan atau akses fisik ke data center. Akses dilakukan menggunakan laptop pribadi melalui koneksi internet biasa[2]. Permintaan data dikirim menggunakan aplikasi Postman sebagai klien REST API[14]. Setelah login menggunakan metode POST untuk memperoleh token autentikasi, peneliti mengirimkan permintaan GET API ke endpoint relevan. Setiap respons yang diterima dalam format JSON disimpan dan didokumentasikan untuk proses verifikasi MBSS[19].

Data konfigurasi yang diperoleh digunakan untuk memeriksa kesesuaian jenis informasi yang ditampilkan oleh sistem *Cisco ACI* dengan kebutuhan verifikasi parameter *Minimum Baseline Security Standard* (MBSS)[3]. Fokus penelitian bukan menilai tingkat keamanan konfigurasi, melainkan memastikan bahwa data hasil GET API merepresentasikan elemen konfigurasi yang relevan berdasarkan dokumen MBSS yang disusun mengacu pada NIST *Special Publication* 800-53[7]. Parameter MBSS meliputi panjang minimal karakter kata sandi, kombinasi karakter, kebijakan riwayat sandi, batas percobaan login gagal, dan pengelolaan akun pengguna[8]. Evaluasi diarahkan pada kesesuaian data terhadap dokumen acuan, bukan pada nilai konfigurasinya[2].

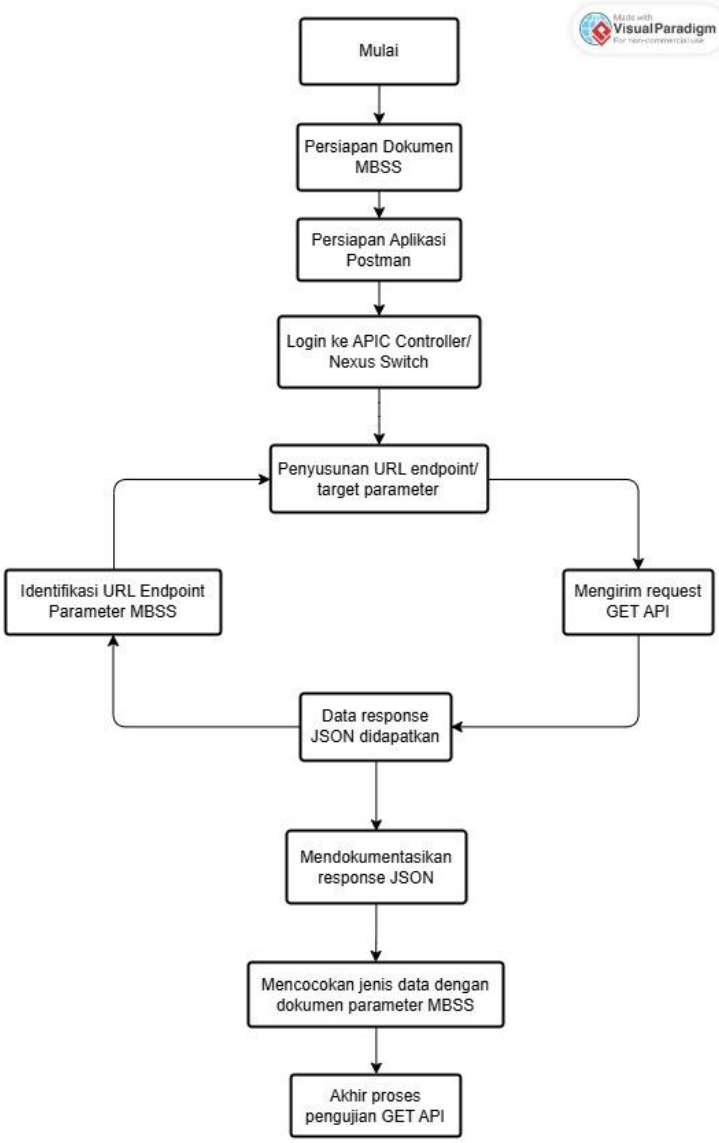
Penelitian ini menggunakan pendekatan eksploratif dengan metode studi kasus. Pendekatan eksploratif dipilih untuk menggali secara mendalam kemampuan metode GET API dalam menampilkan data konfigurasi MBSS pada perangkat *Cisco Nexus ACI*. Ruang lingkup penelitian dibatasi pada evaluasi struktur dan relevansi data yang ditampilkan oleh API terhadap parameter MBSS, bukan pada penilaian tingkat keamanan aktual. Dengan metode studi kasus, pengamatan dilakukan pada sistem nyata yang disimulasikan melalui *Cisco APIC* Sandbox, sehingga hasilnya diharapkan memberikan pemahaman praktis dan aplikatif terkait penerapan API dalam proses verifikasi konfigurasi keamanan jaringan[11].

Tahapan penelitian dilakukan secara sistematis sebagai berikut:

1. Persiapan Dokumen MBSS: Menyusun daftar parameter keamanan mengacu pada NIST SP 800-53 sebagai acuan verifikasi.
2. Akses *Cisco APIC* Sandbox: Menggunakan domain *sandboxapicdc.cisco.com* tanpa koneksi VPN organisasi.
3. Penyiapan Aplikasi *Postman*: Digunakan untuk autentikasi dan pengiriman permintaan GET API.
4. Autentikasi API: Mengirim permintaan POST untuk memperoleh token autentikasi.
5. Penyusunan URL *Endpoint* GET API: Berdasarkan struktur *class*, *managed object*, dan *distinguished name* yang relevan dengan parameter MBSS.
6. Pengiriman Permintaan GET API: Mengambil data konfigurasi yang relevan.
7. Penyimpanan dan Dokumentasi Data: Menyimpan hasil respons JSON untuk proses analisis.
8. Pencocokan Data dengan Dokumen MBSS: Mengevaluasi kesesuaian data yang diperoleh terhadap parameter yang telah ditentukan.

Rancangan pengujian dirancang untuk mengevaluasi sejauh mana metode GET API dapat menampilkan jenis data konfigurasi yang diperlukan untuk verifikasi MBSS pada *Cisco Nexus ACI*. Pengujian dilakukan dengan mengakses setiap parameter MBSS melalui *endpoint* API yang relevan, kemudian membandingkan hasil JSON dengan dokumen MBSS. Kriteria penilaian bersifat biner: Sesuai, jika data konfigurasi yang ditampilkan relevan dan dapat digunakan untuk memverifikasi parameter MBSS. Tidak Sesuai, jika data yang ditampilkan tidak relevan, tidak muncul, atau tidak dapat digunakan untuk verifikasi. Hasil pengujian didokumentasikan dalam tabel yang memuat parameter MBSS, URL *endpoint*, cuplikan respons JSON, dan status kesesuaian data, yang selanjutnya akan dianalisis pada bab pembahasan.

e-ISSN: 2797-5274
p-ISSN: 2809-9788



Gambar 1 Diagram Alur Penelitian

3. HASIL DAN PEMBAHASAN

Sebelum melaksanakan implementasi metode GET API, peneliti terlebih dahulu mempersiapkan lingkungan uji yang mendukung proses pengujian secara terstruktur dan replikatif. Setelah lingkungan pengujian dipersiapkan, langkah selanjutnya adalah melakukan proses autentikasi terhadap *Cisco APIC Controller* melalui API. Setelah berhasil melakukan autentikasi dan memperoleh token login dari *Cisco APIC Sandbox*, peneliti mulai melakukan permintaan GET API untuk menarik data konfigurasi berdasarkan parameter-parameter yang tercantum dalam dokumen MBSS. Setiap permintaan ditujukan ke URL *endpoint* tertentu yang telah disusun sesuai dengan struktur sistem *Cisco ACI* dengan klik SEND. *Endpoint* yang digunakan umumnya memiliki struktur: “https://sandboxapicdc.cisco.com/api/class/<nama-class>.json”

Setiap *name-class* mewakili objek atau kategori konfigurasi tertentu, seperti pengguna (*aaaUser*), profil sandi (*aaaPwdProfile*), hingga parameter lain yang berkaitan dengan kebijakan keamanan akun. Pemilihan class yang tepat sangat krusial agar data yang ditarik benar-benar sesuai dengan parameter MBSS yang ingin diverifikasi. Permintaan dikirim menggunakan metode GET melalui aplikasi *Postman*, dengan token autentikasi yang secara otomatis disisipkan ke dalam setiap permintaan sebagai bagian dari sesi login sebelumnya. Jika *endpoint* dan token valid, maka sistem akan merespons dengan data konfigurasi dalam format JSON yang terdiri dari pasangan key-value. Contoh permintaan *endpoint* yang digunakan antara lain:

e-ISSN: 2797-5274
p-ISSN: 2809-9788

Tabel 1 Target Permintaan *URL Endpoint*

Baseline ID	***Baseline Security Control***	***URL Endpoint***	**Ilustrasi Nilai *Minimum Baseline* (tidak dinilai)**
ID-01	*Show User Accounts*	https://sandboxapicdc.cisco.com/api/class/aaaUser.json?rsp-prop-include=naming-only	*Each authorized individual has their own account*
ID-02	*Minimum password*	https://sandboxapicdc.cisco.com/api/mo/uni/userext.json?query-target=subtree&target-subtree-class=aaaPwdProfile,aaaPwdStrengthProfile	*8 Character*
ID-03	*Password combination*		*uppercase letters, lowercase letters, numbers, and special characters*
ID-04	*Password age (hours)*		*Minimum age 3 days, Maximum age 90 days*
ID-05	*Max password history saved*		*5 passwords*
ID-06	*Account failed login attempt threshold*	https://sandboxapicdc.cisco.com/api/mo/uni/userext/blockloginp.json	*Max failed login attempts 5, Lockout duration minimum 30 minutes*

Hasil respon JSON dari permintaan tersebut menampilkan informasi konfigurasi aktual dalam sistem. Informasi ini kemudian akan dicocokkan dengan parameter MBSS yang relevan untuk menilai apakah data yang ditampilkan memang sesuai dengan yang dibutuhkan. Secara umum, proses GET API ini dilakukan secara berulang untuk setiap parameter MBSS yang berbeda, dengan *endpoint* dan *class* yang disesuaikan. Seluruh data yang berhasil ditarik kemudian disimpan dan diorganisasikan sebagai bagian dari dokumentasi hasil pengujian.

Selanjutnya, setiap hasil respon dari permintaan GET API yang berhasil dieksekusi pada sistem *Cisco ACI* disimpan dan didokumentasikan secara sistematis untuk keperluan analisis dan pencocokan dengan parameter MBSS. Penyimpanan dilakukan dalam dua bentuk utama, yaitu secara digital sebagai file *.json* dan secara visual dalam bentuk tangkapan layar (*screenshot*) pada aplikasi *Postman*. Seluruh respon GET API disimpan dalam format JSON mentah, menggunakan fitur ekspor atau salin langsung dari *Postman*. File ini kemudian dikelompokkan berdasarkan parameter MBSS yang terkait, seperti data akun pengguna, kebijakan sandi, atau batas percobaan login. Dengan metode ini, data hasil pengujian dapat dengan mudah diakses kembali untuk keperluan dokumentasi lanjutan maupun audit internal. Setelah itu akan muncul *pop up* file *manager* untuk dilakukan penyimpanan pada folder yang diinginkan.

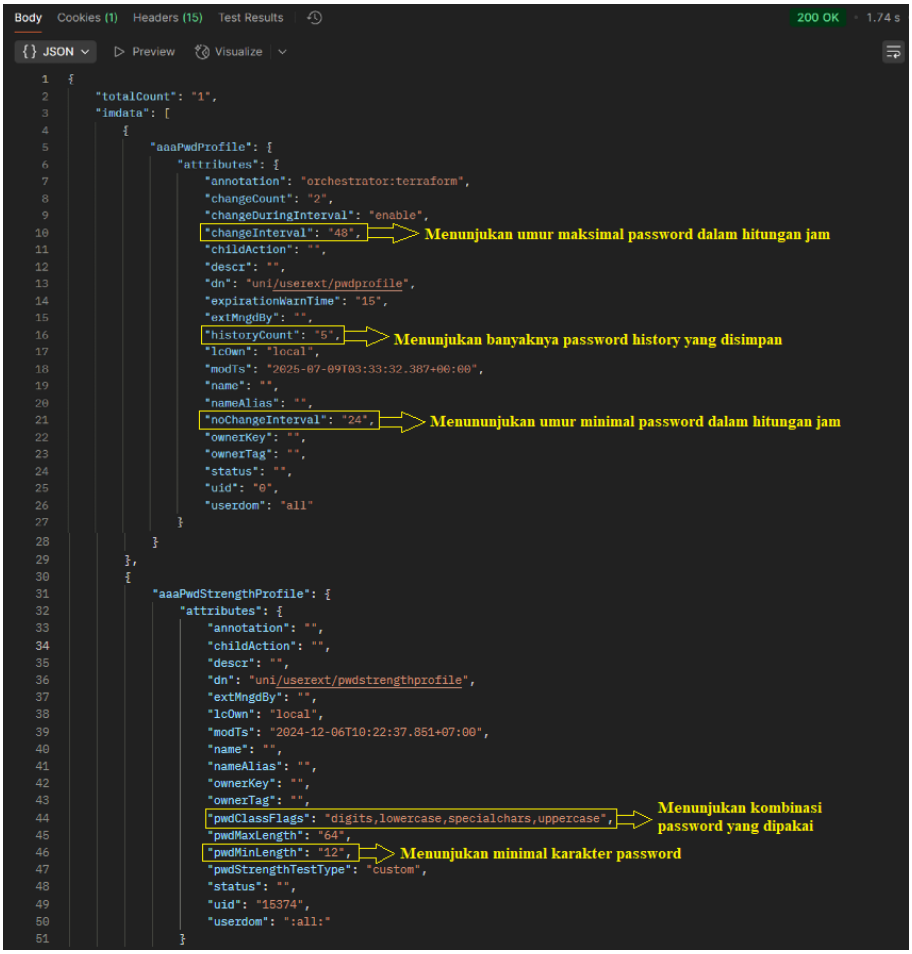
Untuk mendukung keterbacaan dan mempermudah analisis, data JSON yang telah disimpan juga dijelaskan ulang dalam dokumentasi terstruktur yang disusun ke dalam *spreadsheet*. Tabel ini mencantumkan masing-masing parameter MBSS, URL *endpoint* yang digunakan, jenis konfigurasi yang berhasil ditarik, serta apakah data tersebut dapat dianggap sebagai data yang tepat dan relevan. Dengan pendekatan ini, hasil pengujian tidak hanya terekam sebagai bukti teknis, tetapi juga terdokumentasi dengan rapi dalam format yang mudah diverifikasi ulang dan digunakan kembali untuk pengujian pada sesi berikutnya atau pada perangkat serupa lainnya.

Berikutnya, setelah seluruh hasil respon GET API terdokumentasi, peneliti melakukan proses pencocokan terhadap setiap parameter konfigurasi yang tercantum dalam dokumen MBSS. Tujuan pencocokan ini adalah untuk memastikan bahwa data yang ditampilkan melalui respon JSON memang merupakan jenis data yang dibutuhkan dalam proses verifikasi MBSS, tanpa melakukan penilaian terhadap isi atau nilai keamanannya. Dibawah ini adalah bukti pencocokan antara respon data JSON dengan parameter yang relevan dari dokumen MBSS. Pada gambar berikut ditampilkan hasil respon dari *endpoint* aaaUser.json, yang menunjukkan daftar nama pengguna (*user account*) yang terdaftar di dalam sistem. Dua entri teridentifikasi, yaitu "*admin*" dan "*oneaciapp*". Nilai ini berada pada atribut *name* di dalam objek *aaaUser*. Hasil ini tepat dan sesuai dengan parameter MBSS ID-01 yang mengharuskan sistem dapat menampilkan daftar akun pengguna yang aktif.

e-ISSN: 2797-5274
p-ISSN: 2809-9788

Gambar 2 Hasil Respons Konfigurasi *User Accounts*

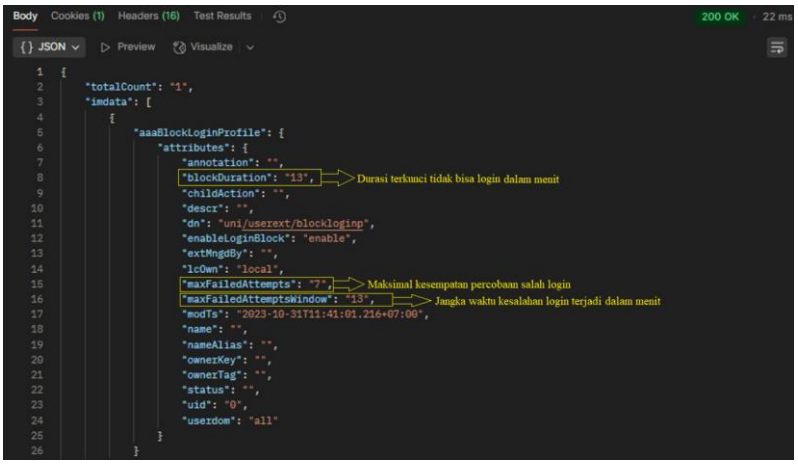
Gambar berikut menunjukkan hasil respon dari *endpoint* userext.json, yang menampilkan konfigurasi profil sandi (*aaaPwdProfile*) dan kekuatan sandi (*aaaPwdStrengthProfile*). Atribut *noChangeInterval* dan *changeInterval* menunjukkan umur minimum dan maksimum sandi (dalam jam). Atribut *historyCount* menunjukkan jumlah sandi sebelumnya yang disimpan. Atribut pwdMinLength dan *pwdClassFlags* menunjukkan minimum karakter sandi dan kombinasi jenis karakter (huruf besar, kecil, angka, simbol). Semua atribut ini secara langsung menjawab parameter MBSS ID-02 sampai ID-05. Oleh karena itu, respon JSON ini dinyatakan tepat dan sesuai dengan jenis data konfigurasi yang dibutuhkan untuk verifikasi MBSS terkait password *policy*.



Gambar 3 Hasil Respons Konfigurasi Password Policy

e-ISSN: 2797-5274
p-ISSN: 2809-9788

Gambar berikut menunjukkan data dari *endpoint blockloginp*.json, khususnya objek *aaaBlockLoginProfile*, yang menampilkan konfigurasi batas upaya login yang gagal. Atribut *maxFailedAttempts* menunjukkan jumlah maksimal percobaan login yang diizinkan. Atribut *maxFailedAttemptsWindow* menunjukkan jangka waktu percobaan kesalahan. Atribut *blockDuration* menunjukkan durasi pemblokiran saat batas kegagalan login terlampaui. Ketiga atribut tersebut menjawab parameter MBSS ID-06 terkait dengan ambang batas upaya login gagal. Hasil ini dapat dikonfirmasi sebagai tepat dan relevan dengan kebutuhan verifikasi MBSS.



Gambar 4 Hasil Respons Konfigurasi Batasan Gagal *Login*

Untuk melengkapi proses pencocokan ini, berikut adalah tabel evaluasi yang merangkum hasil kecocokan jenis data berdasarkan masing-masing parameter MBSS:

Tabel 2 Ringkasan Hasil Pengujian

Baseline ID	***Baseline Security Control***	***URL Endpoint***	**Status Ketepatan Data Yang Didapatkan**
ID-01	*Show User Accounts*	*https://sandboxapicdc.cisco.com/api/class/aaaUser.json?rsp-prop-include=naming-only*	Tepat dan sesuai
ID-02	*Minimum password*	*https://sandboxapicdc.cisco.com/api/mo/uni/userext.json?query-target=subtree&target-subtree-class=aaaPwdProfile,aaaPwdStrengthProfile*	Tepat dan sesuai
ID-03	*Password combination*		Tepat dan sesuai
ID-04	*Password age (hours)*		Tepat dan sesuai
ID-05	*Max password history saved*		Tepat dan sesuai
ID-06	*Account failed login attempt threshold*	*https://sandboxapicdc.cisco.com/api/mo/uni/userext/blockloginp.json*	Tepat dan sesuai

Pada hasil akhir pengujian, berdasarkan seluruh proses implementasi dan pencocokan data yang telah dilakukan, pengujian akhir terhadap metode verifikasi *Minimum Baseline Security Standard* (MBSS) melalui GET API pada sistem *Cisco ACI* dapat disimpulkan berhasil. Proses dimulai dari tahap login dan autentikasi API, dilanjutkan dengan permintaan GET API untuk setiap parameter MBSS yang relevan, kemudian disusul dengan pencocokan terhadap jenis data yang dikembalikan dalam bentuk JSON. Seluruh hasil GET API berhasil menampilkan atribut-atribut konfigurasi keamanan yang sesuai dengan kebutuhan verifikasi MBSS, antara lain: Daftar *username*; Jumlah minimal karakter *password*; Kebijakan kombinasi *password*; Waktu umur *password*; Jumlah *password* yang disimpan; Kebijakan jumlah kesalahan *login* yang diperbolehkan.

Untuk mempermudah pemahaman dan validasi, seluruh hasil data JSON telah didokumentasikan melalui tangkapan layar (*screenshot*) dan dijelaskan satu per satu pada subbab sebelumnya, dengan disertai tabel evaluasi sebagai bentuk formalitas akhir. Dengan demikian, hasil akhir pengujian membuktikan bahwa metode GET API dapat digunakan sebagai pendekatan otomatisasi awal dalam proses verifikasi MBSS, tanpa perlu bergantung pada proses manual melalui CLI atau GUI.

e-ISSN: 2797-5274
p-ISSN: 2809-9788

4. KESIMPULAN

Berdasarkan rangkaian proses penelitian dan implementasi yang telah dilakukan, dapat disimpulkan bahwa tujuan utama penelitian telah tercapai secara tepat dan menyeluruh, serta berhasil menjawab seluruh permasalahan yang diangkat dalam penelitian ini. Penelitian ini membuktikan bahwa metode GET API pada arsitektur *Cisco Nexus ACI* dapat digunakan untuk menarik data konfigurasi MBSS dalam format JSON secara langsung dan terstruktur. Setiap parameter konfigurasi dapat ditampilkan melalui satu langkah permintaan GET pada *endpoint* API yang sesuai. Dengan format JSON yang dihasilkan, informasi menjadi lebih mudah dibaca, didokumentasikan, dan dievaluasi. Ini secara langsung menjawab kebutuhan akan proses verifikasi yang ringkas, efisien, dan tidak tersebar di banyak antarmuka. Lebih lanjut, penelitian ini juga berhasil mengisi kekosongan pendekatan yang secara spesifik memanfaatkan metode GET API untuk kebutuhan verifikasi MBSS, yang sebelumnya belum banyak dibahas baik dalam praktik industri maupun dalam literatur akademik. Berdasarkan hasil pengujian, metode GET API ini berhasil mengimplementasikan verifikasi MBSS dengan tingkat keberhasilan 100%, di mana seluruh 6 dari 6 parameter kontrol MBSS (ID-01 hingga ID-06) yang diuji dapat ditarik dan diverifikasi melalui satu langkah permintaan GET API pada endpoint yang sesuai. Hasil eksplorasi dan dokumentasi dalam penelitian ini menjadi bukti bahwa antarmuka API yang telah tersedia sebenarnya memiliki potensi besar untuk mendukung proses verifikasi konfigurasi keamanan jaringan secara sistematis dan konsisten. Dengan demikian, masalah-masalah utama dalam penelitian ini telah terjawab secara menyeluruh dan sesuai dengan arah tujuan penelitian yang ditetapkan sejak awal. Pendekatan GET API terbukti sebagai metode yang layak, presisi, dan dapat menjadi pijakan awal untuk pengembangan verifikasi MBSS yang lebih modern di masa depan.

5. UCAPAN TERIMA KASIH

Penulis mengucapkan terima kasih kepada Sekolah Tinggi Ilmu Komputer Cipta Karya Informatika atas dukungan selama proses penelitian. Apresiasi juga disampaikan kepada Cisco Systems, Inc. yang telah menyediakan fasilitas Laboratorium Publik *Cisco Nexus ACI* untuk pengujian dan verifikasi data. Ucapan terima kasih diberikan pula kepada para *reviewer* atas saran dan masukan yang membangun terhadap naskah ini.

REFERENSI

[1] H. Taherdoost, “Understanding Cybersecurity Frameworks and Information Security Standards—A Review and Comprehensive Overview,” *Electronics*, vol. 11, no. 14, 2022, doi:10.3390/electronics11142181.
[2] F. Djebbar and K. Nordstrom, “A Comparative Analysis of Industrial Cybersecurity Standards,” *IEEE Access*, vol. 11, no. July, pp. 85315–85332, 2023, doi:10.1109/ACCESS.2023.3303205.
[3] D. Olifer, N. Goranin, A. Cenys, A. Kaceniauskas, and J. Janulevicius, “Defining the minimum security baseline in a multiple security standards environment by graph theory techniques,” *Applied Sciences*, vol. 9, no. 4, 2019, doi:10.3390/app9040681.
[4] S. M. Ali, A. Razzaque, M. Yousaf, and R. U. Shan, “An Automated Compliance Framework for Critical Infrastructure Security through Artificial Intelligence,” *IEEE Access*, vol. 13, pp. 4436–4459, 2024, doi:10.1109/ACCESS.2024.3524496.
[5] D. Mellado, E. Fernandez-Medina, and M. Piattini, “Applying a Security Domain Requirements Engineering Process for Software Product Lines,” *IEEE Latin America Transactions*, vol. 6, no. 3, pp. 298–305, 2008, doi:10.1109/tla.2008.4653861.
[6] S. Majumdar et al., “User-Level Runtime Security Auditing for the Cloud,” *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 5, pp. 1185–1199, 2018, doi:10.1109/TIFS.2017.2779444.
[7] S. P. Ong et al., “The Materials Application Programming Interface (API): A simple, flexible and efficient API for materials data based on REpresentational State Transfer (REST) principles,” *Computational Materials Science*, vol. 97, pp. 209–215, 2015, doi:10.1016/j.commatsci.2014.10.037.
[8] K. N. Markert et al., “Design and implementation of a BigQuery dataset and application programmer interface (API) for the U.S. National Water Model,” *Environmental Modelling & Software*, vol. 179, p. 106123, 2024, doi:10.1016/j.envsoft.2024.106123.
[9] F. Palma, T. Olsson, A. Wingkvist, and J. Gonzalez-Huerta, “Assessing the linguistic quality of REST APIs for IoT applications,” *Journal of Systems and Software*, vol. 191, p. 111369, 2022, doi:10.1016/j.jss.2022.111369.
[10] P. Bourhis, J. L. Reutter, and D. Vrgoč, “JSON: Data model and query languages,” *Information Systems*, vol. 89, 2020, doi:10.1016/j.is.2019.101478.

e-ISSN: 2797-5274
p-ISSN: 2809-9788

[11] C. O. Truică, E. S. Apostol, J. Darmont, and T. B. Pedersen, “Oriented Database Management Systems: An Overview and Benchmark of Native XML DODBMSes in Comparison with JSON DODBMSes,” *Big Data Research*, vol. 25, 2021, doi:10.1016/j.bdr.2021.100205.
[12] R. Maurya, “Application of Restful APIs in IoT: A Review,” *International Journal for Research in Applied Science and Engineering Technology*, vol. 9, no. 2, pp. 145–151, 2021, doi:10.22214/ijraset.2021.33013.
[13] D. Shah and V. Giomo, “Verified Scalability Guide for Cisco APIC, Release 6.1(3) and Cisco Nexus 9000 Series ACI-Mode Switches, Release 16.1(3),” *Text. View Mag.*, vol. 1, no. 113, pp. 184–185, 2016.
[14] B. Lebiednik, A. Mangal, and N. Tiwari, “A Survey and Evaluation of Data Center Network Topologies,” 2016, [Online]. Available: http://arxiv.org/abs/1605.01701.
[15] P. Ijari, “Comparison between Cisco ACI and VMWARE NSX,” *IOSR Journal of Computer Engineering*, vol. 19, no. 01, pp. 70–72, 2017, doi:10.9790/0661-1901047072.
[16] E. Chavarriaga, F. Jurado, and F. D. Rodríguez, “An approach to build JSON-based Domain Specific Languages solutions for web applications,” *Journal of Computer Languages*, vol. 75, p. 101203, 2023, doi:10.1016/j.cola.2023.101203.
[17] Y. H. Wang and I. C. Wu, “Definition of REST web services with JSON schema,” *Software: Practice and Experience*, vol. 39, no. 7, pp. 701–736, 2009, doi:10.1002/spe.
[18] J. S. Horsburgh, K. Lippold, and D. L. Slaugh, “Adapting OGC’s SensorThings API and data model to support data management and sharing for environmental sensors,” *Environmental Modelling & Software*, vol. 183, p. 106241, 2025, doi:10.1016/j.envsoft.2024.106241.
[19] Y. Ding, Z. Wu, Z. Tan, and X. Jiang, “Research and application of security baseline in business information system,” *Procedia Computer Science*, vol. 183, pp. 630–635, 2021, doi:10.1016/j.procs.2021.02.107.
[20] NIST SP800-53, *Security and Privacy Controls for Information Systems and Organizations*. Gaithersburg, MD, 2020, doi:10.6028/NIST.SP.800-53r5.