Deteksi Serangan SQL *Injection* Menggunakan *Security Information and Event Management* (SIEM) Wazuh (Sudi Kasus: Sistem Informasi Akademik Universitas Mataram)

Detection of SQL Injection Attacks Using Security Information and Event Management (SIEM) Wazuh (Case Study: Academic Information System University of Mataram)

Novianda Shafira Suryawatie Yomo[1], Ahmas Zafrullah Mardiansyah[1], I Wayan Agus Arimbawa[2]
[1] Program Studi Teknik Informatika, Universitas Mataram
Jl. Majapahit 62, Mataram, Lombok NTB, INDONESIA
Email: noviandafiraa@gmail.com, [zaf, arimbawa] @unram.ac.id

Indonesia currently needs a national cybersecurity strategy in the era of Society 5.0. If security is defined as freedom from threats or dangers, then one of the most important drivers in managing cybersecurity is how threats are understood in cyberspace and then finding solutions for them. Without proper efforts to enhance cybersecurity, the likelihood of threats increasing is high. There are many types of attacks on websites, and one of the most common types in Indonesia is SQL Injection. The National Cyber and Encryption Agency (BSSN) recorded that SQL Injection attacks accounted for 73% of attacks in Indonesia from January to April 2019. SQL Injection technique exploits security vulnerabilities in a web server. The result of this attack is that the attacker gains access to the database, which can lead to the leakage of personal and institutional data. One system security that can assist in monitoring and analyzing incoming threats to the server is known as Security Information and Event Management (SIEM). SIEM is a monitoring system that detects attacks and responses of a security system through real-time analysis of logs from various data sources. The performance percentage of Wazuh in detecting SQL Injection attacks is 45.67%.

Kata Kunci: SIEM, SQL *Injection*, Wazuh, Keamanan Siber, Sistem Informasi Akademik

I. PENDAHULUAN

Indonesia saat ini membutuhkan strategi keamanan siber nasional era *society* 5.0. Jika keamanan sebagai kebebasan dari ancaman atau bahaya, maka salah satu pendorong yang terpenting dalam mengelola *cyber security* adalah bagaimana ancaman dipahami dalam ruang siber kemudian dicari solusinya. Tanpa upaya meningkatkan keamanan siber yang tepat, kemungkinan ancaman akan meningkat. Tantangan terbesar saat ini adalah penguatan lembaga keamanan siber dan kurangnya tenaga ahli serta kerjasama di dalam negeri maupun dengan dunia internasional. Oleh karena itu, penting bagi pemerintah untuk memperkuat keamanan siber dan mempersiapkan tenaga ahli yang dibutuhkan di era yang semakin digital. Pada era internet saat ini, informasi sangat mudah diperoleh dan disebarluaskan. Informasi menjadi aset yang sangat berharga baik bagi pemerintah. Keamanan informasi dibuat untuk mendapatkan kerahasiaan, ketersediaan, serta integritas pada semua sumber daya informasi perusahaan bukan hanya perangkat keras dan data [1].

Penyimpanan data terletak pada *database* yang biasa digunakan oleh pemilik dan pengembang *website* berupa data *login*, data *user*, dan lain hal sebagainya [2]. Terdapat banyak jenis serangan pada *website*, salah satunya jenis serangan pada *website* yang paling sering terjadi di Indonesia yaitu SQL *Injection*. Badan Siber dan Sandi Negara (BSSN) mencatat serangan SQL *Injection* yang terjadi di Indonesia dari bulan Januari s/d April 2019 sebanyak 73% [3]. Teknik SQL *Injection* sudah dikenal dalam dunia *hacking* sebagai salah satu Teknik *web hacking* dan dapat merusak *database* pada suatu sistem. Teknik SQL *Injection* memanfaatkan celah keamanan pada suatu *webserver*. Hasil yang ditimbulkan dari serangan ini, penyerang akan mendapatkan akses ke *database* yang dapat menyebabkan kebocoran data pribadi maupun data suatu instansi [4]. Adapun contoh kasus nyata serangan SQL *Injection*, seperti yang diberitakan oleh CNN Indonesia Data Pengguna Tokopedia Bocor ke Dark Web. Data tersebut dijual seharga US$5.000 (Rp74,3 juta) di sebuah situs jual beli dark web. Sebanyak 91 juta data pengguna dan lebih dari tujuh juta data *merchant* di *e-commerce* dibocorkan oleh *hacker* bernama ShinyHunters [5].

Berkaitan dengan keamanan siber yang menjadi isu prioritas di berbagai negara termasuk Indonesia, setiap instansi perlu meningkatkan keamanan pada sistem guna menghindari potensi yang dapat merugikan instansi

bahkan negara [6]. Dalam suatu sistem terdapat banyak aktivitas pertukaran data, sehingga untuk memelihara data penting dalam suatu sistem diperlukan sebuah sistem keamanan dan pemeliharaan sistem yang baik. Salah satu instansi yang menerapkan teknologi informasi adalah Universitas Mataram. Penggunaan teknologi informasi di Universitas Mataram memungkinkan terjadinya proses pembuatan, penyimpanan, dan sarana berbagai pengetahuan termasuk pada bidang akademik. Sistem Informasi Akademik (SIA) Universitas Mataram merupakan salah satu contoh teknologi informasi yang digunakan sebagai sarana untuk menyimpan data akademik mahasiswa.

Sistem Informasi Akademik (SIA) Universitas Mataram pernah mengalami peretasan, seperti yang diberitakan oleh Inside Lombok pada tahun 2019. Mahasiswa yang ingin melihat nilai merasa kebingungan ketika mengakses SIA dikarenakan yang ditampilkan memuat unsur-unsur politik. Kepala Pusat Teknologi Informasi dan Komunikasi (Pustik) Unram, Cahyo Mustiko, menerangkan bahwa peretasan memang berpotensi terjadi di situs Unram [7]. Akibat dari serangan SQL *Injection* yang pernah terjadi beberapa tahun lalu menyebabkan *hacker* dapat menembus *database* hingga merugikan instansi, dosen dan mahasiswa.

Untuk mencegah serangan SQL *Injection* dan memastikan pelayanan pada Sistem Informasi Akademik (SIA) dapat berjalan dengan baik dan aman maka diperlukan upaya untuk meningkatkan sistem keamanan pada SIA Unram. Salah satu keamanan sistem yang dapat membantu dalam memonitoring dan menganalisa ancaman yang masuk ke dalam *server* salah satunya dikenal dengan *Security Information and Event Management* (SIEM). SIEM merupakan sistem monitoring yang mampu mendeteksi serangan dan respon suatu sistem keamanan melalui analisis log dari berbagai kejadian yang berasal dari berbagai sumber data secara *realtime* [8]. Teknologi SIEM memiliki jangkauan pengumpulan data yang luas serta dapat mengkorelasikan dan menganalisis kejadian dari berbagai sumber dan menentukan apakah kejadian tersebut merupakan suatu serangan atau tidak [9].

Pada penelitian ini, untuk mendeteksi serangan SQL *Injection* pada Sistem Informasi Akademik (SIA) Universitas Mataram menggunakan Wazuh sebagai SIEM. Wazuh digunakan karena dapat mengolah data dengan skala besar serta pengembangan yang cepat dan pemeliharaan yang mudah. Selain itu Wazuh memiliki *interface* yang mudah dipahami oleh pengguna. *Interface* Wazuh berupa grafik, sehingga dapat dengan mudah dipahami oleh *administrator* untuk melakukan *monitoring*. Pada Wazuh *dashboard* juga terdapat data seberapa serangan yang masuk, *agent* mana yang paling sering aktif dan jenis serangan yang paling sering terjadi pada Wazuh *agent* [10].

II. TINJAUAN PUSTAKA

Penelitian terkait metode pengujian serangan SQL *Injection* telah banyak dilakukan sebagai bahan evaluasi perbaikan sistem selanjutnya. Dalam kurun waktu sepuluh tahun terakhir terdapat berbagai penelitian sebelumnya yang akan dijadikan sebagai bahan rujukan dalam penelitian ini.

Pada penelitian yang berjudul "Deteksi Anomali dengan *Security Information and Event Management* (SIEM) Splunk pada Jaringan UII" yang telah dilakukan oleh M. Rijal Kamar pada tahun 2021. Penelitian ini bertujuan sebagai bentuk peningkatan sistem keamanan pada teknologi informasi UII dengan bantuan keamanan SIEM yaitu Splunk. Dari penelitian yang telah dilakukan bahwa deteksi anomaly pada *log firewall* jaringan UII berhasil dilakukan dengan menerapkan aturan yang sudah dibuat, akan tetapi pengujian masih sebatas sebagian *log firewall* dan belum utuh dalam satu hari penuh atau membandingkan dengan hari lainnya [9].

Pada penelitian yang berjudul "Detection of DOS *Attack* and *Zero-Day Threat with* SIEM" yang telah dilakukan oleh Sornalakshmi pada tahun 2017. Penelitian ini membahas Dos dan Zero-Day Attack. SIEM digunakan sebagai alat pemantauan untuk memantau informasi yang dikumpulkan pada *server*. Cara kerja SIEM yaitu dengan *Logs Collect*, Normalize and Correlate, Analyze, dan *Store* and *Report*. Dos *Attack* merupakan salah satu jenis serangan yang langsung menargetkan infrastruktur *server* dan mengeksploitasi kerentanan dalam aplikasi yang bertujuan untuk membaca *webserver*, mencuri data rahasia dan mengancam pengguna. Sedangkan *Zero-Day Attack* merupakan serangan yang biasanya mencuri data-data yang berupa identitas atau informasi lainnya pada website yang nantinya data-data yang dicuri akan dijual kepada pihak ketiga. Dengan mendeteksi DoS *Attack* menggunakan access logs pada Apache *webserver* yang menjadi inti pada penelitian ini. *Rule Implementation of* SIEM akan menjadi tempat untuk memeriksa semua logs yang telah dikumpulkan. Sedangkan untuk mendeteksi *Zero-Day Attack*, penulis mengasumsikan pengguna menggunakan Windows *Operating System* dan mengumpulkan *log* dari 'Aplication Logs', 'System Logs' dan 'Security Logs' yang akan dikirim lagi ke SIEM dan diperiksa mengguna Rule Implementation Zero-Day Attack. Hasil penelitian membuktikan bahwa jika berhasil menggunakan Rule sebagai parameter pemantauan, akan dapat mendeteksi serangan DoS dengan *log* yang telah didapatkan dan dapat menghasilkan peringatan untuk hal yang berpotensi menyebabkan kerusakan pada server. *Tool* ini dengan mudah diintegrasikan pada *server*. Sama halnya dengan *Zero-Day Threat*, untuk mencegah segala kerugiaan dan mencegah terjadinya pada sistem jika dieksploitasi, dapat menggunakan konsep SIEM untuk mendeteksi serangan DoS dan serangan Zero-Day *Attack* [11]

Pada penelitian yang berjudul "Implementasi Sistem Pendeteksi Serangan SQL *Injection* dengan Menggunakan Algoritma K-*Nearest Neighbor*" yang telah dilakukan oleh

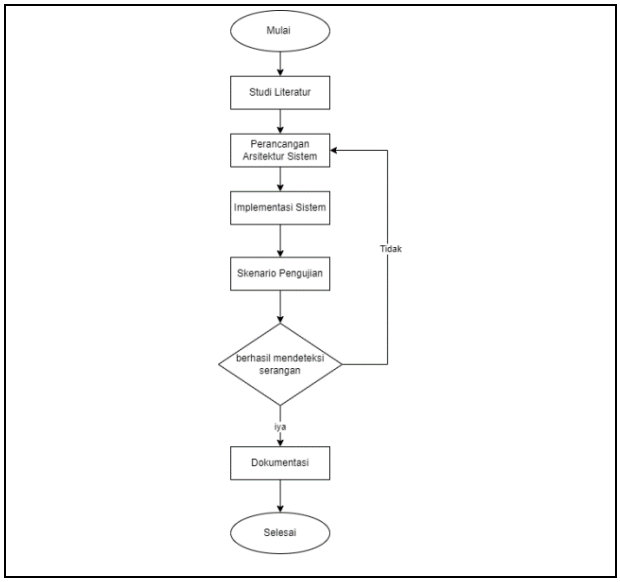
Rangga Dinata pada tahun 2019. Penelitian ini mendeteksi serangan SQL *Injection* dengan menggunakan metode deteksi anomali dengan algoritma K-*Nearest Neighbor* karena dapat mengklasifikasi data berdasarkan jarak terdekat dari data yang diuji ke data latih. Sehingga data yang masuk akan di hitung besar kemiripan datanya dengan data latih berdasarkan jarak terpendek dan akan di klasifikasikan menjadi serangan atau bukan. Hasil dari penelitian itu membuktikan bahwa sistem pendeteksi serangan SQL *Injection* dapat membedakan data aman dan data serangan melalui tahapan Pre-*processing* dan klasifikasi. Sistem IDS membutuhkan mekanisme Pre-*processing* untuk mengolah data mentah menjadi siap untuk diproses oleh classifier. Hasil pendeteksian menunjukan bahwa tingkat akurasi tertinggi dengan menggunakan 13.895 data latih sebesar 100%. Hasil klasifikasi tersebut menghasilkan data yang diprediksi sebagai '*malicious'* lebih banyak dibanding data 'legit'. Pada penggunaan algoritme KNN semakin besar data latih yang digunakan oleh sistem maka akan semakin baik tingkat akurasi yang akan didapatkan [12]

Berdasarkan tinjauan pustaka yang telah dilakukan, pada penelitian sebelumnya pendeteksian serangan SQL *Injection* dilakukan menggunakan algoritma KNN yang belum dapat secara optimal dalam memilih jenis data dan jumlah terbaiknya. Selain itu penelitian tersebut tidak dapat menampilkan serangan yang terjadi secara *realtime* dikarenakan pengujian sebelumnya hanya sampai pengujian akurasi dengan menggunakan metode KNN. Pada penelitian ini juga menggunakan Wazuh sebagai SIEM merupakan salah satu aplikasi *Open Source* yang baru namun telah memiliki fitur yang lengkap.

III. METODE PENELITIAN

Pada tahapan penelitian ini meliputi Alur penelitian mengenai Deteksi Serangan SQL *Injection* Menggunakan *Security Information and Event Management* terhadap Sistem Informasi Akademik Universitas Mataram.

A. Alur Penelitian



Gambar 1. Diagram Alir Penelitian

TABLE 1. ALUR PENELITIAN

Tahap	Penjelasan
Studi literatur	Merupakan tahap awal pada penelitian ini yaitu dilakukan dengan mencari referensi pada jurnal-jurnal terkait dengan peneltiain seperti SQL *Injection*, Algoritma Serangan SQL *Injection*, Implementasi Security Information and Event Management, dan Sistem Informasi Akademik (SIA)
Perancangan Arsitektur	Tahap pembuatan konsep perancangan arsitektur seperti bagaiman alur kerja dari sistem. perancangan ini diawali dengan membuat Wazuh *Manager* pada *host* atau server Universitar mataram dilanjutkan dengan pemasangan *agent*
Implementasi Sistem	Implementasi sistem dilakukan pembuatan sistem berdasarkan dengan konsep yang telah dirancang.
Skenario Pengujian	Pengujian dilakukan serangan SQL *Injection* pada sistem dengan menggunakan Burp Suite sebagai tools pengujian.
Dokumentasi	Dokumentasi berupa laporan dari hasil penelitian yang telah dilakukan.

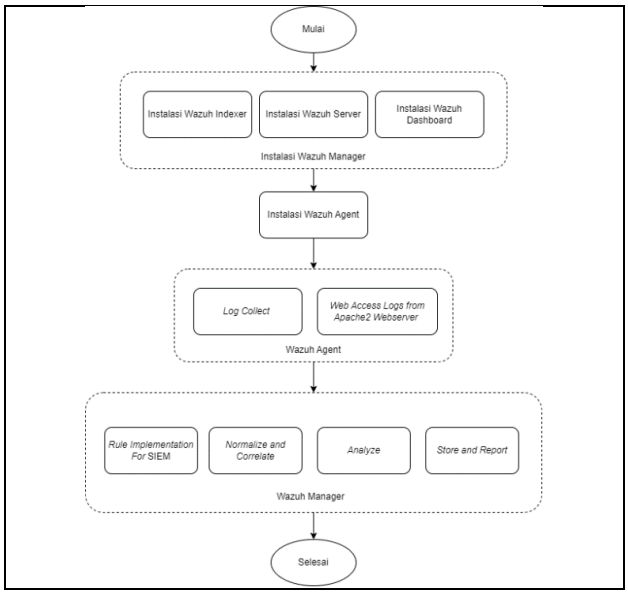
B. Analisis Kebutuhan

Pada tahapan analisis kebutuhan meliputi beberapa *software* dan *hardware* yang digunakan dalam penelitian sebagai berikut:

TABLE II. SOFTWARE DAN HARDWARE

Software	Hardware
Wazuh *version* 4.3	Laptop Asus Vivobook S14
Web browser	
SSH Terminal	
Ubuntu *server version* 22.04	
Burp Suite *version* 12.7	

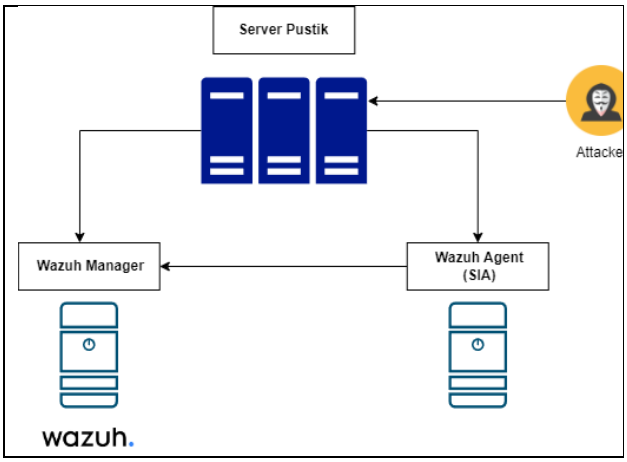
C. *Perancangan Sistem*



Gambar 2. Diagram Perancangan Sistem

1. Instalasi Wazuh *indexer* merupakan komponen utama Wazuh dalam melakukan pengindeksan dan menyimpan peringatan yang dihasilkan oleh Wazuh *server*.
2. Instalasi Wazuh *server* untuk menganalisis data yang diterima dari Wazuh *agent*, memicu peringatan saat terdapat ancaman atau anomali yang terdeteksi. Selain itu juga digunakan untuk mengelola konfigurasi *agent* secara *remote* dan *monitoring* status *agent*.
3. Instalasi Wazuh *dashboard* merupakan *interface* pada Wazuh yang menampilakan visual dari data. Wazuh *dashboard* dapat mempermudah pengguna karena dapat menampilkan visualisasi peristiwa keamanan, grafik kerentanan yang terdeteksi.
4. Instalasi Wazuh *agent*, pada tahap ini dilakukan instalasi Wazuh *agent* pada SIA Unram.
5. *Log Collect*, data *log* yang diambil merupakan sejumlah data besar yang diambil dari berbagai sumber dan tipe *log* yang berbeda.
6. *Web Access Logs from Apache2 Webserver*, untuk mendeteksi serangan SQL *Injection* diperoleh dari *log* akses pada *apache* 2 *webserver* yang akan langsung masuk kedalam SIEM.
7. *Rule Implementation For* SIEM, merupakan bagian implementasi aturan yang dimana semua *log* akan diperiksa dengan aturan yang telah diterapkan.
8. *Normalize and Correlate*, data yang telah didapatkan kemudian disortir untuk mengidentifikasi hubungan dan pola guna mendeteksi serta merespons potensi ancaman dengan cepat.
9. *Analyze*, sistem mengolah dan menganalisis data dari berbagai sumber secara *realtime*.
10. *Store and Report*, hasil dari *analyze* data *log* akan dilaporkan dan ditampilkan dalam bentuk grafik pada Wazuh *dashboard*.

D. Topologi Jaringan

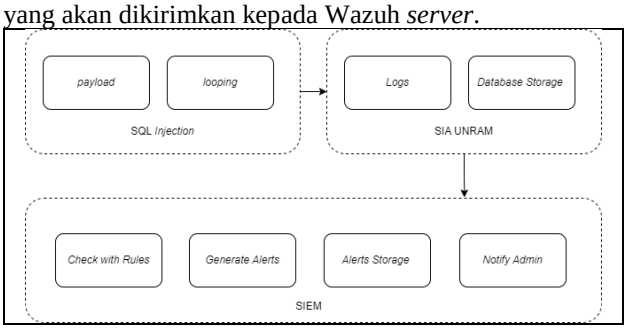


Gambar 3. Topologi Jaringan

Pada Gambar 3 terdapat topologi jaringan pada pemasangan dari sistem yang akan dibuat. Pemasangan Wazuh *manager* dan Wazuh *agent* diinstal pada *server* Pustik yang menjadi *server* utama. *Attacker* melakukan serangan jenis SQL *Injection* pada *server* utama yaitu *server* Pustik. *Agent* yang terdapat pada *host server* SIA akan mengirimkan *log* atau informasi serangan pada Wazuh *manager*. *Alert* yang ditangkap dapat dilihat dan divisualiasi pada *dashboard* Wazuh *manager*.

E. Skenario Pengujian

Pada proses pengujian, SIEM akan dipasang pada *host* Universitas Mataram kemudian dilakukan pemeriksaan *host* apakah memiliki sumber daya yang cukup untuk menjamin kinerja yang tepat. Setelah dilakukan pemeriksaan *host*, Wazuh *agent* akan dipasang pada perangkat *endpoint* yaitu SIA Universitas Mataram untuk dilakukan pembacaan sistem, pengumpulan *log* yang akan dikirimkan kepada Wazuh *server*.



Gambar 3. *Flowchart* Skenario Pengujian

1. Pengujian dilakukan dengan melakukan serangan SQL *Injection* pada server SIA dengan memasukan *payload* SQL *Injection* menggunakan *Burp Suite* sebagai *Penetration Testing*.
2. Dilakukan pengujian secara berulang dengan menggunakan *payload* yang berbeda, diantaranya dengan menggunakan tipe *In-Band* SQL *Injection* dan *Inferensial* SQL *Injection*.

3. Wazuh *agent* yang telah terpasang pada SIA akan menangkap *log* serangan yang telah dilakukan dan akan disimpan kedalam *database*.
4. Tahap selanjutnya dengan memeriksa semua *log* terhadap aturan yang diterapkan. Jika aturan cocok, akan menghasilkan peringatan.
5. Seluruh peringatan akan disimpan pada *database* dan pemantauan *log* akan terus dilakukan.

Data akan divisualisasi pada Wazuh *dashboard* yang akan mempermudah *administrator* dalam melakukan analisis sistem dan berbagai masalah keamanan lainnya.

F. Dokumentasi laporan

Pada tahapan ini dilakukan setelah melakukan pengujian sistem dilanjutkan pada tahap dokumentasi dan laporan. Hasil dari pengujian akan didokumentasikan dan diambil kesimpulan berdasarkan pengujian yang telah dilakukan. Kesimpulan yang telah didapatkan akan dapat dignakan sebagai refrensi dalam penelitian selanjutnya.

IV. HASIL DAN PEMBAHASAN

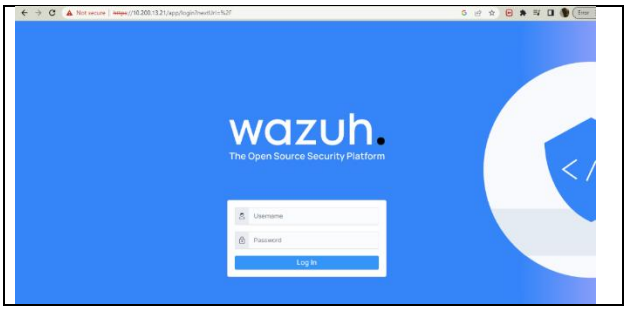
Pada Bab ini akan membahas hasil dan penelitian yang dilakukan yaitu mengimplementasikan SIEM sebagai "Deteksi Serangan SQL *Injection* Menggunakan *Security Information and Event Management* (SIEM) Wazuh (Studi Kasus: Sistem Informasi Akademik Universitas Mataram). Pembuatan sistem dilakukan sesuai dengan perancangan yang telah dijabarkan pada Bab sebelumnya. Pembahasan akan meliputi instalasi Wazuh *manager*, Instalasi Wazuh *agent*, Pengujian serangan SQL *Injection*, dan Analisis *Log* pada Wazuh.

A. Instalasi Wazuh Manager

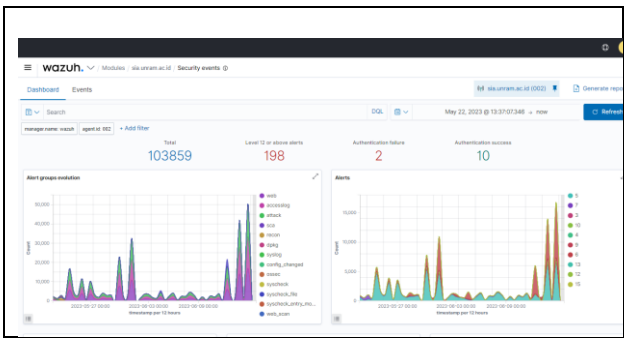
Wazuh *manager* merupakan aplikasi *open source* yang berfungsi sebagai sistem deteksi berbasis *host* (*endpoint*). Wazuh dapat memonitoring aktivitas yang dilakukan oleh Wazuh *agent* dan kemudian *log* tersebut divisualisasikan oleh Wazuh dengan beragam bentuk grafik yang mudah dipahami. Berikut langkah-langkah dalam memasang Wazuh *manager* sebagai berikut:

1. Menyiapkan *host server* pada *server* Universitas Mataram. Pada implementasi penelitian ini menggunakan sistem operasi Ubuntu Server 22.04. Selanjutnya melakukan instalasi dan konfigurasi pada Wazuh manager pada OS Ubuntu server yang sudah diinstal.
2. Melakukan instalasi Wazuh indexer, Wazuh server dan Wazzuh dashboard. Proses instalasi dilakukan dengan menggunakan satu perintah sebagai berikut:

```
curl-sO
https://packages.wazuh.com/4.3/wazu
h-install.sh
```

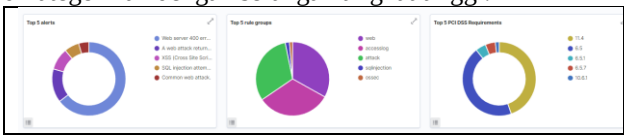


Gambar 4. Tampilan Login pada Wazuh



Gambar 4. *Dashboard* Wazuh

Pada Gambar 4 merupakan *dashboard* yang terdapat pada Wazuh *manager*. Pada halaman ini terdapat beberapa informasi dari *log* yang ditangkap oleh Wazuh, seperti total *log* yang telah ditangkap oleh Wazuh dari aplikasi Sistem Informasi Akademik (SIA) yaitu sebanyak 103859 *log*, banyaknya *log* yang memiliki level 12 atau lebih sebanyak 198 *log*, jumlah autentikasi gagal dan autentikasi berhasil. *Log* yang ditangkap diklasifikasikan ke dalam beberapa level yaitu level 1 hingga level 15. Level 1 hingga 6 dikategorikan dengan serangan tingkat rendah, level 7 hingga level 11 dikategorikan dengan serangan tingkat menengah dan level 12 hingga 15 dikategorikan dengan serangan tingkat tinggi.



Gambar 5. *Dashboard Security Event*

Pada Gambar 5 merupakan beberapa grafik yang menampilkan 5 data tertinggi dari beberapa kategori seperti Top 5 *alerts*, Top 5 *rule groups* dan Top 5 PCI DSS *Requirement*.

B. Instalasi Wazuh Agent

Wazuh *agent* merupakan perangkat yang diinstal pada perangkat *endpoint* untuk melakukan pembacaan sistem, pengumpulan *log* serta mengirimkan pada Wazuh *server*. Berikut langkah-langkah instalasi Wazuh *agent* pada aplikasi SIA Unram:

1. Menginstall Wazuh agent pada perangkat *endpoint* SIA Unram.

2. Menginstall *package* yang diperlukan untuk proses instalasi Wazuh *agent* menggunakan perintah sebagai berikut:

```
“apt install curl apt-transport-
    https unzip wget libcap2-bin
    software-properties-common
    lsb-release gnupg”.
```

3. Menginstal kunci GPG sebagai proses enkripsi data menggunakan perintah berikut:

```
“curl                          -s
    https://packages.wazuh.com/key
    /GPG-KEY-WAZUH | gpg --no-
    default-keyring --keyring
    gnupg-
    ring:/usr/share/keyrings/wazuh
    .gpg --import && chmod 644
    /usr/share/keyrings/wazuh.gpg”
```

4. Menambahkan *repository* Wazuh pada aplikasi SIA dengan menggunakan perintah sebagai berikut:

```
“echo "deb [signed-
    by=/usr/share/keyrings/wazuh.g
    pg]
    https://packages.wazuh.com/4.x
    /apt/ stable main" | tee -a
    /etc/apt/sources.list.d/wazuh.
    list”
```

5. Menginstall wazuh agent ke dalam sistem menggunakan perintah berikut:

```
WAZUH_MANAGER= “10.200.xx.xx” apt-
    get install wazuh-agent”
```

6. Mengaktifkan Wazuh agent dengan perintah berikut:

```
“systemctl daemond-reload”
    “systemctl enable wazuh-agent”
    “systemctl start wazuh-agent”.
```

C. Pengujian Serangan SQL *Injection*

Pada langkah ini, peneliti bertindak sebagai pengunjung biasa yang tidak memiliki akun pada sistem dan penguji menyimulasikan diri sebagai penyerang. Berikut merupakan langkah-langkah pengujian yang dilakukan sebagai berikut:

1. *Vulnerability Scanning* dengan menggunakan Burp Suite

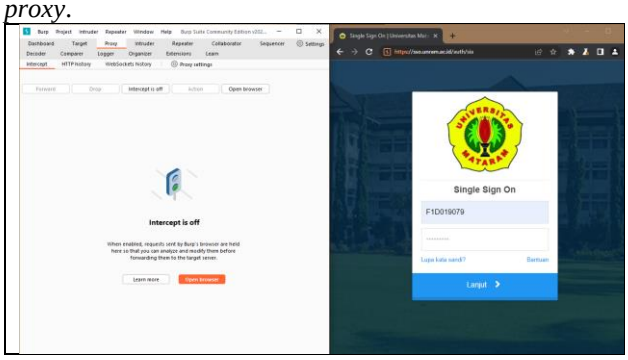
Vulnerability Scanning merupakan proses pengujian keamanan yang bertujuan untuk mengidentifikasi kerentanan dalam sistem, jaringan atau aplikasi. Langkah pertama dalam skenario *vulnerability scannimg* dengan Burp Suite yaitu menentukan situs *website* yang akan dijadikan target. Pada penelitian ini yang akan dijadikan target adalah Sistem Informasi Akademik (SIA) Universitas Mataram. Kemudian proses dilanjutkan dengan melakukan scanning dengan *tools* pada Burp Suite. Berikut langkah-langkah sebagai berikut:

- Menentukan Target

Pada penelitian ini menggunakan aplikasi Sistem Infomasi Akademik (SIA) Unram sebagai target pengujian. Aplikasi ini dapat diakses melalui sia.unram.ac.id pada *browser* yang terdapat pada Burp Suite. Ketika alamat target dieksekusi pada *browser* dengan proxy sudah mengarah ke aplikasi Burp Suite, maka *proxy* Burp Suite akan mengintrupsi prosesnya dengan mengarahkan ke *proxy* Burp Suite.

- Halaman login target

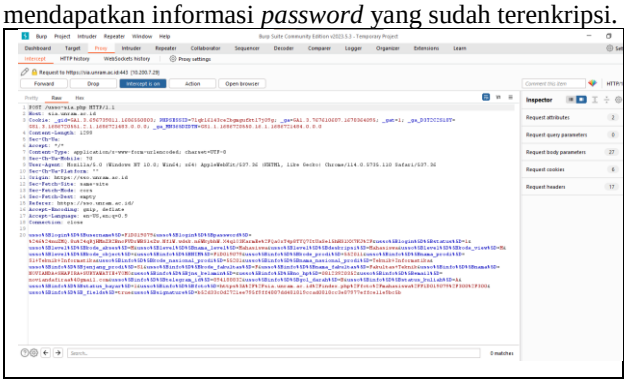
Pada halaman login SIA yang akan digunakan untuk melakukan *bruteforce* didapatkan dengan memasukan *username* dan *password* pada *form login* SIA. Pada proses ini akan menghasilkan beberapa eksekusi yang terekam pada Burp Suite dan dapat terlihat pada *history* HTTP *proxy*.



Gambar 5. Halaman Login Target

- Meneruskan (forward) pada Burp Suite

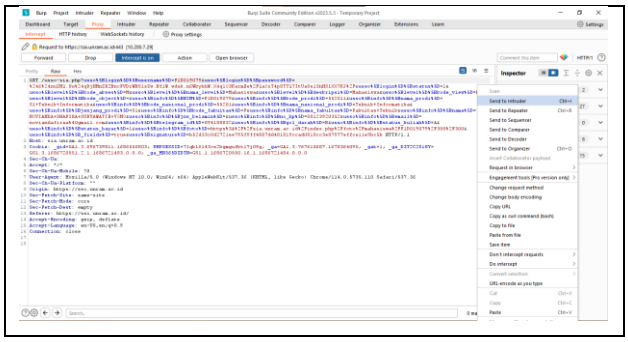
Pada Gambar 4.17 merupakan proses meneruskan (*forward*) semua *request* yang di dapat dari *browser* untuk mendapatkan informasi *password* yang sudah terenkripsi.



Gambar 6. Proses *forward* pada Burp Suite

- Mengirimkan Data ke Menu Intruder

Fitur *Intruder* digunakan untuk melakukan serangan otomatis dan menguji kelemahan keamanan pada *website*. Setelah memilih permintaan, klik kanan pada permintaan tersebut dan pilih opsi “*Send to Intruder*”.



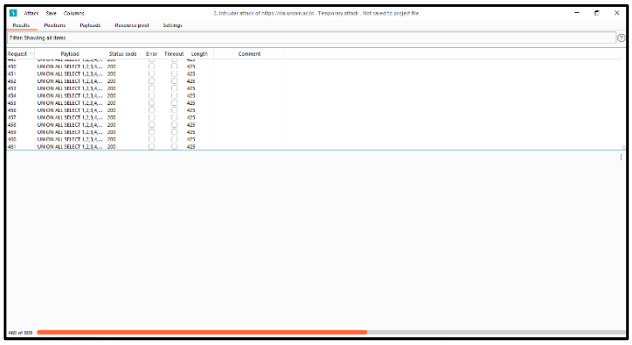
Gambar 7. Proses *send to intruder*

Gambar 8. Menentukan Posisi data untuk dimanipulasi

Pada Gambar 8 setelah data dikirim ke menu *intruder* akan terlihat beberapa bagian seperti "*Positions*", "*Payloads*", dan "*Attacks Type*". Pada bagian "*Positions*", menentukan bagian dari data yang ingin dimanipulasi atau diganti dengan *payloads*, pada bagian yang ingin dimanipulasi diberikan simbol dolar ($) sebagai penanda bagian yang akan dieksekusi. Pada bagian *positions* juga terdapat "*Attack Type*" yang dapat memilih jenis serangan yang akan dilakukan pada penelitian ini menggunakan tipe jenis *sniper*.

Gambar 9. Memasukan *payloads* SQL injection

Pada bagian "*Payloads*", dapat menentukan *payloads* yang ingin digunakan dalam serangan. *Payloads* dapat berupa nilai yang bervariasi atau kombinasi yang berbeda untuk menguji berbagai skenario. Setelah memilih *payloads* yang ingin digunakan, pada bagian pojok kanan atas terdapat "*Start Attack*" yang digunakan untuk memulai serangan otomatis menggunakan *payloads* yang telah ditentukan.



Gambar 10. Proses menginjeksi payload SQL Injection

Pada Gambar 10 merupakan proses menginjeksi *payloads* SQL *Injection*. Terdapat 808 *payloads* yang dimasukan.

D. Hasil Pengujian

Dalam mendeteksi serangan SQL *Injection*, Wazuh mendeteksi serangan SQL *Injection* dengan menggunakan beberapa metode dan Teknik. Berikut merupakan cara Wazuh dalam mendeteksi serangan SQL *Injection*:

1. Wazuh memantau dan menganalisis *log* dari berbagai sumber, termasuk *server web*, aplikasi dan basis data. Wazuh mencari pola serangan yang umum digunakan dalam SQL *Injection*, seperti karakter tanda kutip yang tidak valid, tanda-tanda operasi logika SQL yang mencurigakan atau upaya injeksi pernyataan SQL lainnya. Jika pola-pola ini ditemukan dalam *log*, Wazuh menghasilkan peringatan untuk menunjukan adanya serangan SQL *Injection*.
2. Wazuh menggunakan Teknik deteksi anomali untuk mengeali perilaku yang tidak biasa dalam *query* SQL. Deteksi ini melinatkan pemantauan dan analisis pola kueri yang biasanya dilakukan oleh aplikasi atau pengguna. Jika ada penyimpangan dari pola yang signifikan pada pola *query* yang normal, Wazuh dapat menghasilkan peringatan untuk menandakan kemungkinana adanya serangan SQL *Injection*.
3. Wazuh memantau dan memeriksa parameter HTTP, termasuk parameter dalam URL dan data yang dikirimkan melalui metode POST. Wazuh mencari karakteristik serangan SQL *Injection*, seperti penggunaan karakter tanda kutip yang tidak valid atau upaya injeksi pernyataan SQL lainnya. Jika ada tanda-tanda serangan SQL *Injection* pada parameter-parameter ini, Wazuh dapat menghasilkan peringatan.

Pengujian dilakukan dengan cara mempersiapkan list *payloads* yang akan digunakan dalam penyerangan. List *payloads* ini didapatkan pada github yang memiliki *review* pemakaian paling banyak digunakan. Selanjutnya, proses memasukkan *payloads* tersebut pada aplikasi Burp Suite dan memilih menu *payload* pada tab *intruder*, pada menu tersebut terdapat payload *set* dan payload *options*. Payload *Set* digunakan untuk mengatur *text* yang akan digunakan

pada *form* halaman *login*. Setelah itu list *payloads* dimasukin ke dalam *form payloads options*.

Berdasarkan total jumlah payload yang telah diujikan dengan jumlah total serangan yang terdeteksi sebagai SQL Injection berjumlah 13 dengan level yang terdeteksi yaitu level 6, hal ini dikarenakan jika serangan SQL Injection berhasil dieksekusi dan mencapai backend database, maka aturan deteksi SQL Injection yang sesuai dapat aktif dan menghasilkan alert SQL Injection dan serangan yang terdeteksi sebagai Web Attack Returned 200 dengan level yang terdeteksi yaitu level 6 berjumlah 356 hal ini dikarenakan serangan SQL Injection tidak berhasil dieksekusi secara penuh atau tidak mencapai backend database, namun aplikasi masih memberikan respons dengan status code 200, maka hasil deteksi yang muncul berupa Web Attack Returned 200. Serangan ini juga menunjukan bahwa permintaan yang dikirimkan berhasil diproses oleh server tanpa ada kesalahan yang terdeteksi pada protokol HTTP, meskipun serangan SQL Injection tidak terdeteksi secara langsung, respons dengan status code 200 ini memungkinkan memberikan petunjuk bahwa ada potensi kerentanan dalam aplikasi yang memungkinkan serangan SQL Injection dilakukan. Dari hasil pengujian yang dilakukan didapatkan perhitungan persentase kinerja Wazuh dalam mendeteksi serangan SQL Injection sebagai berikut:

$$\frac{369}{808} \times 100\% = 45{,}67\%$$

Berdasarkan hasil persentase yang didapatkan, penggunaan 808 *payload* untuk serangan SQL *Injection* dan terdeteksi hanya 369 serangan yang oleh Wazuh, sisa serangan *payload* dapat tidak terdeteksi atau dapat terdeteksi oleh metode deteksi yang lain. Terdapat beberapa alasan mengapa sisa serangan *payload* tidak terdeteksi, diantaranya sebagai berikut:

1. Rule dan Konfigurasi: Wazuh menggunakan aturan deteksi yang telah dikonfigurasi untuk mendeteksi serangan SQL *Injection*. Apabila ada payload yang digunakan tidak cocok dengan pola atau karakteristik yang ada dalam aturan deteksi, maka serangan tidak akan terdeteksi. Kemungkinan bahwa *payload* yang tidak terdeteksi oleh Wazuh tidak sesuai dengan pola yang diatur dalam aturan deteksi SQL *Injection* yang digunakan.

2. Kelemahan dalam Aturan Deteksi: Setiap aturan deteksi memiliki keterbatasan dan tidak dapat mendeteksi semua jenis serangan SQL *Injection*. Aturan deteksi yang digunakan oleh Wazuh dapat diperbarui dan disesuaikan untuk meningkatkan deteksi terhadap serangan SQL *Injection* yang lebih kompleks. Apabila *payload* yang digunakan teknik atau pola yang belum diperbarui dalam aturan deteksi, maka serangan tersebut mungkin tidak terdeteksi. Adapun beberapa aturan pada Wazuh dalam mendeteksi serangan SQL *Injection* sebagai berikut:

TABLE III. RULE ID DALAM MENDETEKSI SQL INJECTION

Rule ID	Deskripsi	Level
31106	Deteksi serangan yang berpotensi SQL *Injection*	6
31107	Deteksi upaya serangan SQL *Injection*	6
31164	Deteksi percobaan SQL *Injection* yang diterima	6
31205	Kemungkinan upaya SQL *Injection*	6
31210	Upaya yang dicurigai sebagai SQL *Injection*	6

3. Variasi *Payload*: Terdapat banyak variasi payload yang dapat digunakan *dalam* serangan SQL *Injection*. Dari jumlah 808 *payload*, masih terdapat kemungkinan bahwa variasi *payload* yang digunakan belum terdeteksi oleh Wazuh. Hal ini terkait dengan keberagaman dan kompleksitas serangan SQL *Injection* yang dapat dilakukan oleh penyerang.

V. KESIMPULAN DAN SARAN

A. Kesimpulan

Berdasarkan penelitian dan pengujian yang telah dilakukan, dapat ditarik kesimpulan sebagai berikut:

1. Pada aplikasi Sistem Informasi Akademik (SIA) telah diimplementasikan *Security Information and Event Management* (SIEM) yang dapat memonitoring sistem keamanan pada SIA itu sendiri. Dengan menggunakan SIEM, *administrator* dapat mengumpulkan dan melakukan pemantauan secara *real time* serta mengalisis *log* untuk mendeteksi aktivitas mencurigakan yang mengancam keamanan.
2. Wazuh yang digunakan sebagai implementasi SIEM menyediakan *interface* agar visualisasi dalam proses memonitoring dan mengalisis *log* serta keamanan terjadi dengan baik.
3. Berdasarkan hasil pengujian yang dilakukan menggunakan Wazuh untuk mendeteksi serangan SQL *Injection* pada Sistem Informasi Akademik, ditemukan bahwa dari total 808 kombinasi payloads yang diuji, terdapat 369 serangan yang dikelompokan sebagai serangan SQL *Injection* diantaranya terdeteksi sebagai *web attack returned code* 200 (*success*) dengan jumlah 356 peringatan dan 13 peringatan yang terdeteksi sebagai SQL *Injection*. Hal ini menunjukkan kinerja Wazuh dalam mendeteksi serangan SQL *Injection* sebesar 45,67%. Sisa serangan payload yang tidak terdeteksi dapat disebabkan oleh beberapa faktor, seperti

ketidakcocokan *payload* dengan pola atau karakteristik dalam aturan deteksi, kelemahan dalam aturan deteksi yang tidak dapat mendeteksi semua jenis serangan SQL *Injection*, atau penggunaan teknik atau pola serangan yang belum diperbarui dalam aturan deteksi yang digunakan.

B. Saran

Sebagai upaya untuk mengembangkan hasil yang didapatkan dalam penelitian ini agar menjadi lebih baik, beberapa saran yang dapat diberikan adalah sebagai berikut:

1. Untuk pengembangan sistem selanjutnya, perlu dilakukan analisis terhadap fitur lain pada Wazuh selain pada *security event*.
2. Mendeteksi serangan lain yang berpotensi kritikal pada Sistem Informasi Akademik (SIA) Universitas Mataram.

DAFTAR PUSTAKA

[1] Budi Eko, Wira Dwi, and Infantono Ardian, "Strategi Penguatan Cyber Security Guna Mewujudkan Keamanan Nasional di Era Society 5.0," Prosiding Seminar Nasional Sains Teknologi dan Inovasi Indonesia (SENASTINDO), vol. 3, pp. 223–234, Dec. 2021, doi: 10.54706/senastindo.v3.2021.141.

[2] A. Saputra, H. H. Kusuma, and A. Ibrahim, "Mengatasi Keamanan di dalam SQL Injection dan Cara Pencegahannya," Prosiding Annual Research Seminar 2017 Computer Sceince and ICT, vol. 3, pp. 105–108, 2017.

[3] Badan Siber dan Sandi Negara, "MENGENAL SQL INJECTION DAN CARA MENCEGAHNYA."

[4] E. T. Arizki, "INVESTIGASI FORENSIK SERANGAN WEBSITE SQL INJECTION MENGGUNAKAN METODE SCRIPTING PADA CRAWLTRACK," UPN Veteran, Jawa Timur, 2022.

[5] A. Wixaksono, "Penelusuran 91 Juta Data Bocor Tokopedia, Dijual Rp74 Juta," CNN Indonesia, May 03, 2020.

[6] R. Christianingrum and A. N. Aida, "Tantangan Penguatan Keamanan Siber dalam Menjaga Stabilitas Keamanan," 2021.

[7] B. Pratama, "SIA Unram Diretas, PSI Merasa Dirugikan," Inside Lombok, Mataram, Jan. 09, 2019.

[8] C. Arfanudin, B. Sugiantoro, and Y. Prayudi, "ANALISIS SERANGAN ROUTER DENGAN SECURITY INFORMATION AND EVENT MANAGEMENT (SIEM) DAN IMPLIKASINYA PADA INDEKS KEAMANAN INFORMASI," CyberSecurity dan Forensik Digital, vol. 2, pp. 1–7, May 2019.

[9] M. Rijal Kamal and M. Andri Setiawan, "Deteksi Anomali dengan Security Information and Event Management (SIEM) Splunk pada Jaringan UII".

[10] M. D. Pratama, F. Nova, and D. Prayama, "Wazuh sebagai Log Event Management dan Deteksi Celah Keamanan pada Server dari Serangan Dos," Jurnal Ilmiah Teknologi Sistem Informasi JITSI, vol. 3, pp. 1–7, Mar. 2022.

[11] Sornalakshmi, "Detection of DoS attack and Zero Day Threat with SIEM," International Conference on Intelligent Computing and Control Systems ICICCS, pp. 1–7, 2017.

[12] R. Dinata, "Implementasi Sistem Pendeteksi Serangan SQL Injection dengan Menggunakan Algoritme K-Nearest Neighbor," Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer, vol. 13, Dec. 2019.

[13] A. Anggono and M. Riskiyadi, "Cybercrime dan Cybersecurity pada Fintech: Sebuah Tinjauan Pustaka Sistematis Cybercrime and Cybersecurity at Fintech: A Systematic Literature Review," Jurnal Manajemen dan Organisasi (JMO), vol. 12, no. 3, pp. 239–251, 2021.

[14] A. D. Djayali, "Analisa Serangan SQL Injection pada Server pengisian Kartu Rencana Studi (KRS) Online," Jurnal Ilmiah Manajemen Informatika & Komputer, Sep. 2020, doi: 10.14710/gt.v21i4.46828.

[15] A. Subari, S. Manan, E. Ariyanto, and F. Adnan, "PEMANFAATAN METODE WAVS (WEB APPLICATION SECURITY SCANNERS) MENGGUNAKAN BURP SUITE TOOLS DALAM AUDIT TEKNIS KEAMANAN SISTEM INFORMASI SURAT TUGAS SEKOLAH VOKASI UNDIP," GEMA TEKNOLOGI, vol. 21, no. 4.

[16] Citra Arfanudin, B. Sugiantoro, and Y. Prayudi, "ANALISIS SERANGAN ROUTER DENGAN SECURITY INFORMATION AND EVENT MANAGEMENT (SIEM) DAN IMPLIKASINYA PADA INDEKS KEAMANAN INFORMASI ANALYSIS OF ROUTER ATTACK WITH SECURITY INFORMATION AND EVENT MANAGEMENT AND IMPLICATIONS (SIEM) IN INFORMATION SECURITY INDEX," 2019.

[17] M. Rijal Kamal and M. Andri Setiawan, "Deteksi Anomali dengan Security Information and Event Management (SIEM) Splunk pada Jaringan UII."

[18] J. Chandra, "IMPLEMENTASI SISTEM INFORMASI AKADEMIK (STUDI KASUS: SMP NEGERI 20 BANDUNG)."