Journal of Artificial Intelligence and Technology Information
***Website* : https://ejournal.techcart-press.com/index.php/jaiti**
E-ISSN 2985-6396
P-ISSN 2985-5306
Volume 4, Number 2, Juni 2026
https://doi.org/10.58602/jaiti.v4i2.262
Page 287-299

Deteksi Serangan *Web* Berdasarkan *Audit Log ModSecurity* Menggunakan *Support Vector Machine* (SVM)

Nuroji[1*], Tirta Anhari[2]
[1]Teknik Informatika, Universitas Muhammadiyah Prof. DR. HAMKA, Indonesia
[2]Sistem dan Tekno*logi* Informasi, Universitas Muhammadiyah Prof. DR. HAMKA, Indonesia
[1*]nuroji@uhamka.ac.id, [2]tirta.anhari@uhamka.ac.id

Abstrak: Perkembangan aplikasi web yang semakin pesat meningkatkan risiko serangan siber terhadap layanan berbasis internet, sementara *Web Application Firewall* (*WAF*) berbasis *rule-based* seperti *ModSecurity* masih memiliki keterbatasan dalam mengenali variasi pola serangan baru serta menghasilkan bias pada data tidak seimbang. Penelitian ini bertujuan untuk mengklasifikasikan traffic web dari log *ModSecurity* menggunakan pendekatan *machine learning* berbasis *Support Vector Machine* (*SVM*) yang dikombinasikan dengan *TF-IDF* dan penanganan data tidak seimbang menggunakan *SMOTE*. *Dataset* penelitian terdiri dari 3467 data traffic web dengan empat kelas, yaitu *SQL Injection* (*SQLi*), Normal, *Local File Inclusion* (*LFI*), dan *Remote Code Execution* (*RCE*). Metode penelitian meliputi *preprocessing*, *labeling*, *feature extraction TF-IDF*, *SMOTE*, *train-test split*, pemodelan *SVM*, dan evaluasi menggunakan *confusion matrix* serta metrik *accuracy*, *precision*, *recall*, dan *F1-score*. Hasil pengujian menunjukkan model *SVM* memperoleh *accuracy* sebesar 98,12% dengan performa sangat baik pada kelas mayoritas, namun masih terdapat variasi pada kelas dengan jumlah data kecil. Hasil sebelum *SMOTE* menunjukkan ketidakseimbangan data yang signifikan, yaitu *SQLi* (1889), Normal (839), *LFI* (38), dan *RCE* (7), sedangkan setelah *SMOTE* seluruh kelas pada data *training* menjadi seimbang masing-masing sebanyak 1889 data. Hasil penelitian ini menunjukkan bahwa kombinasi *SVM* dan *SMOTE* efektif dalam meningkatkan kemampuan deteksi serangan web berbasis log *WAF*, serta berpotensi menjadi solusi alternatif dalam sistem keamanan web yang lebih adaptif.
Kata Kunci: *Web Application Firewall; ModSecurity; Support Vector Machine; SMOTE; Intrusion Detection System*

Abstract: The rapid growth of web applications has increased cybersecurity threats against internet-based services, while rule-based Web Application Firewalls (WAF) such as ModSecurity still have limitations in recognizing variations of new attack patterns and tend to produce bias when handling imbalanced data. This study aims to classify web traffic from ModSecurity logs using a machine learning approach based on Support Vector Machine (SVM), combined with TF-IDF feature extraction and SMOTE for handling class imbalance. The dataset consists of 3,467 web traffic records with four classes, namely SQL Injection (SQLi), Normal, Local File Inclusion (LFI), and Remote Code Execution (RCE). The research methodology includes preprocessing, labeling, TF-IDF feature extraction, SMOTE, train-test split, SVM modeling, and evaluation using a confusion matrix along with accuracy, precision, recall, and

Nuroji: * Penulis Korespondensi

Journal of Artificial Intelligence and Technology Information
***Website* : https://ejournal.techcart-press.com/index.php/jaiti**
E-ISSN 2985-6396
P-ISSN 2985-5306
Volume 4, Number 2, Juni 2026
https://doi.org/10.58602/jaiti.v4i2.262
Page 287-299

F1-score metrics. The experimental results show that the SVM model achieves an accuracy of 98.12%, with excellent performance on majority classes, although some variations are observed in minority classes. Before applying SMOTE, the dataset was highly imbalanced, consisting of SQLi (1,889), Normal (839), LFI (38), and RCE (7), whereas after SMOTE, all classes in the training set were balanced to 1,889 samples each. The findings indicate that the combination of SVM and SMOTE effectively improves web attack detection based on WAF logs and can serve as a promising alternative for more adaptive web security systems.
Keywords: Web Application Firewall; ModSecurity; Support Vector Machine; SMOTE; Intrusion Detection System

1. PENDAHULUAN

Perkembangan aplikasi *web* yang pesat di era transformasi digital menyebabkan meningkatnya ancaman keamanan siber terhadap layanan berbasis *web*. Berbagai jenis serangan seperti *SQL Injection, Cross-Site Scripting (XSS), Remote Code Execution (RCE)*, dan aktivitas *bot* berbahaya masih menjadi ancaman utama yang dapat mengganggu keamanan aplikasi *web*. Selain itu, kasus *web defacement* dan pencurian data (*data breach*) menunjukkan bahwa aplikasi *web* menjadi salah satu target utama eksploitasi keamanan oleh pihak tidak bertanggung jawab. Kondisi tersebut menyebabkan kebutuhan terhadap proses deteksi dan analisis serangan *web* menjadi semakin penting untuk menjaga keamanan sistem dan data pengguna.

Salah satu mekanisme keamanan yang umum digunakan untuk melindungi aplikasi *web* adalah *Web Application Firewall (WAF),* seperti *ModSecurity*[1]. *ModSecurity* bekerja dengan cara memonitor, memfilter, dan mencatat aktivitas *request HTTP* berdasarkan aturan *(rule-based* detection*)* yang telah ditentukan sebelumnya[2]. Melalui mekanisme tersebut, *ModSecurity* mampu menghasilkan *audit log* yang berisi informasi aktivitas *request, payload, header HTTP*, serta indikasi serangan yang terdeteksi pada aplikasi *web*. *Audit log* tersebut dapat dimanfaatkan sebagai sumber informasi untuk melakukan analisis pola serangan *web*.

Meskipun pendekatan *rule-based* pada *WAF* efektif dalam mendeteksi pola serangan yang telah dikenal, metode ini masih memiliki beberapa keterbatasan. Sistem sangat bergantung pada *signature* dan aturan yang telah tersedia sehingga kurang adaptif terhadap variasi payload maupun pola serangan baru yang terus berkembang. Selain itu, konfigurasi rule yang terlalu ketat dapat menyebabkan *false positive*, sedangkan *rule* yang terlalu longgar berpotensi menghasilkan *false negative*[3]. Keterbatasan tersebut menunjukkan bahwa pendekatan *rule-based* saja belum cukup untuk mendukung proses analisis serangan *web* secara optimal, sehingga diperlukan pendekatan tambahan yang mampu melakukan identifikasi pola serangan berdasarkan karakteristik data *log* keamanan *web*[4].

Salah satu pendekatan yang dapat digunakan adalah penerapan *machine learning* untuk melakukan klasifikasi terhadap *audit log* keamanan *web*. Dalam penelitian ini, metode *Support Vector Machine (SVM)* dipilih karena memiliki kemampuan yang baik dalam menangani data teks berdimensi tinggi hasil ekstraksi fitur menggunakan metode *Term Frequency–Inverse Document Frequency (TF-IDF)*[5]. Selain itu, *SVM* dikenal efektif dalam proses klasifikasi teks karena mampu membentuk *hyperplane* optimal untuk memisahkan kategori data normal dan serangan. Dibandingkan beberapa metode klasifikasi lain seperti *Naive Bayes* dan *Decision Tree, SVM* memiliki kemampuan generalisasi yang baik pada *dataset* berbasis teks dengan jumlah fitur yang besar dan mampu memberikan performa klasifikasi yang stabil[6].

Beberapa penelitian sebelumnya telah menerapkan machine learning untuk mendeteksi serangan jaringan maupun keamanan *web*. Namun, sebagian besar penelitian masih

Nuroji: * Penulis Korespondensi

Journal of Artificial Intelligence and Technology Information
***Website* : https://ejournal.techcart-press.com/index.php/jaiti**
E-ISSN 2985-6396
P-ISSN 2985-5306
Volume 4, Number 2, Juni 2026
https://doi.org/10.58602/jaiti.v4i2.262
Page 287-299

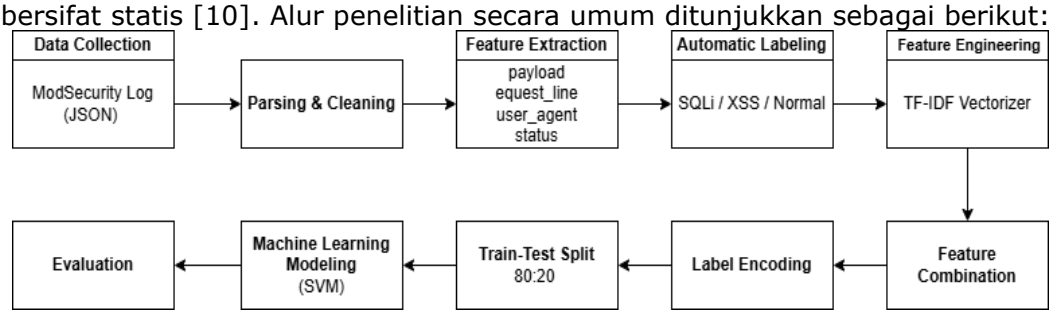
menggunakan *dataset* publik atau data simulasi sehingga belum sepenuhnya merepresentasikan kondisi *log* keamanan pada lingkungan nyata[3]. Selain itu, penelitian sebelumnya umumnya lebih berfokus pada analisis *traffic* jaringan secara umum dan belum secara khusus memanfaatkan *audit log ModSecurity* sebagai sumber data utama dalam proses deteksi serangan *web*[7]. Oleh karena itu, diperlukan penelitian yang memanfaatkan *audit log* nyata dari *Web Application Firewall* untuk mendukung proses analisis dan klasifikasi serangan *web* yang lebih relevan terhadap kondisi aktual.

Kebaruan pada penelitian ini terletak pada penggunaan *audit log ModSecurity* sebagai sumber data utama untuk proses deteksi serangan *web* menggunakan metode Support Vector Machine (SVM)[8]. Penelitian ini berfokus pada analisis *audit log* hasil deteksi *ModSecurity* melalui tahapan parsing *log* JSON, preprocessing data, ekstraksi fitur menggunakan *TF-IDF*, serta proses klasifikasi untuk mengidentifikasi pola *request* normal dan serangan berdasarkan karakteristik *request HTTP*.

Penelitian ini memberikan kontribusi berupa pendekatan deteksi serangan *web* berbasis machine learning menggunakan *audit log ModSecurity* sebagai *dataset* keamanan nyata. Penelitian ini juga menunjukkan bahwa metode *Support Vector Machine (SVM)* dapat digunakan untuk membantu proses analisis *audit log* keamanan *web* dalam mengidentifikasi pola serangan berdasarkan karakteristik *request HTTP* secara lebih terstruktur dan adaptif dibandingkan pendekatan *rule-based* konvensional.

2. METODE PENELITIAN

Penelitian ini menerapkan pendekatan kuantitatif dengan metode eksperimen untuk mengevaluasi kemampuan algoritma *machine learning* dalam mengklasifikasikan *traffic* web ke dalam kategori normal dan serangan. Tahapan penelitian meliputi proses pengumpulan data, *preprocessing*, *labeling*, *feature engineering*, pemodelan, hingga evaluasi model[9]. Seluruh rangkaian tahapan tersebut dirancang untuk menghasilkan model yang mampu melakukan klasifikasi otomatis terhadap data audit log, sekaligus meminimalisir ketergantungan pada teknik pencocokan pola tradisional yang bersifat statis [10]. Alur penelitian secara umum ditunjukkan sebagai berikut:



Gambar 1: Alur penelitian

Data Collection

Tahap awal penelitian dimulai dengan pengumpulan data *log* dari *ModSecurity* yang tersimpan dalam format *JSON*. *Data log* ini berisi informasi aktivitas *traffic web, request HTTP, response server, payload request*, serta hasil deteksi serangan dari *OWASP Core Rule Set (CRS)*[11]. *Dataset* yang digunakan berasal dari *traffic* aktual pada *web server* sehingga mampu merepresentasikan kondisi serangan yang nyata sejumlah 3467 *record* aktivitas *traffic web*.

Preprocessing

Pada tahap ini dilakukan proses *parsing* untuk mengekstraksi informasi penting dari struktur *JSON ModSecurity*. Selain itu dilakukan proses cleaning untuk membersihkan data

Nuroji: * Penulis Korespondensi

Journal of Artificial Intelligence and Technology Information
***Website* : https://ejournal.techcart-press.com/index.php/jaiti**
E-ISSN 2985-6396
P-ISSN 2985-5306
Volume 4, Number 2, Juni 2026
https://doi.org/10.58602/jaiti.v4i2.262
Page 287-299

yang tidak lengkap, duplikat, maupun data yang tidak relevan agar *dataset* lebih siap digunakan pada proses *machine learning*[12].

Feature Extraction

Tahap *feature extraction* bertujuan mengambil atribut-atribut penting dari *log* yang memiliki keterkaitan dengan pola serangan *web*. Beberapa fitur yang digunakan antara lain *payload request*, *request* line, user-agent, status code, dan content-length[13]. Fitur-fitur tersebut digunakan sebagai representasi karakteristik *traffic web*.

Labeling

Pada tahap ini data diberikan label secara otomatis berdasarkan pesan deteksi yang dihasilkan oleh *ModSecurity*. *Traffic* yang mengandung indikasi serangan seperti *SQL Injection* atau *XSS* diberi *label attack*, sedangkan *traffic* tanpa indikasi serangan diberi label normal. Proses *labeling* dilakukan untuk membentuk *dataset* klasifikasi *supervised learning*[14].

Feature Engineering Extraction

Feature engineering dilakukan dengan mengubah data berbentuk teks menjadi representasi numerik yang dapat diproses oleh algoritma *machine learning*[15]. Pada penelitian ini, metode *TF-IDF (Term Frequency–Inverse Document Frequency)* digunakan untuk memberikan bobot pada kata-kata yang terdapat pada *payload request, request line,* dan *user-agent* sehingga model dapat mengenali karakteristik pola serangan dengan lebih baik.

Feature Combination

Tahap ini bertujuan menggabungkan fitur teks hasil *TF-IDF* dengan fitur numerik seperti status *code* dan *content-length*[16]. Penggabungan fitur dilakukan untuk memberikan informasi yang lebih komprehensif kepada model selama proses pembelajaran. Adapun perhitungan *TF-IDF* dapat dinyatakan dengan rumus berikut:

$$TF(t,d) = \frac{f_{t,d}}{\sum_k f_{k,d}} \quad (1)$$

$$IDF(t) = log\left(\frac{N}{df(t)}\right) \quad (2)$$

$$TF\ IDF(t,d) = TF(t,d)x\ IDF(t) \quad (3)$$

Keterangan:

- *TF-IDF*(t,d): nilai bobot term t pada dokumen d
- TF(t,d): frekuensi kemunculan term pada dokumen
- IDF(t): tingkat kepentingan term terhadap keseluruhan dokumen
- t: term atau kata
- d: dokumen

Label Encoding

Pada tahap *label encoding*, label kategori seperti *SQLi*, *XSS*, dan *Normal* diubah menjadi bentuk numerik. Proses ini diperlukan karena algoritma *machine learning* hanya dapat memproses data dalam format numerik[17].

Train-Test Split

Selanjutnya, *dataset* dibagi menjadi data *training* dan data *testing* menggunakan metode stratified split dengan perbandingan 80:20[18]. *Data training* digunakan dalam proses pelatihan *model machine learning*, sedangkan data testing dimanfaatkan untuk mengevaluasi performa model terhadap data baru yang sebelumnya belum pernah dipelajari.

Nuroji: * Penulis Korespondensi

Journal of Artificial Intelligence and Technology Information
***Website* : https://ejournal.techcart-press.com/index.php/jaiti**
E-ISSN 2985-6396
P-ISSN 2985-5306
Volume 4, Number 2, Juni 2026
https://doi.org/10.58602/jaiti.v4i2.262
Page 287-299

SMOTE Oversampling

Peneliti melakukan SMOTE (Synthetic Minority Over-sampling Technique) untuk menangani ketidakseimbangan kelas dalam dataset dengan cara menciptakan sampel sintetis pada kelas minoritas, sehingga model dapat mempelajari karakteristik serangan secara lebih seimbang [19], [20].

Machine learning Modeling

Pada tahap ini, proses pelatihan model *machine learning* dilakukan menggunakan algoritma *Support Vector Machine (SVM).* Model dilatih untuk mengenali pola *traffic* normal maupun *traffic* serangan berdasarkan fitur-fitur yang telah diekstraksi pada tahap sebelumnya[21].

Evaluation Model

Tahap evaluasi dilakukan untuk menilai performa model klasifikasi yang telah dibangun. Proses evaluasi menggunakan beberapa metrik, yaitu *accuracy*, *precision*, *recall*, *F1-score*, dan *confusion matrix*[22]. Hasil pengujian tersebut digunakan untuk mengukur kemampuan model dalam mendeteksi serangan *web* berdasarkan data *log ModSecurity*.

- *Accuracy*
  Accuracy digunakan untuk mengetahui tingkat ketepatan model dalam melakukan klasifikasi terhadap seluruh data secara keseluruhan.

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (4)$$

- *Precision*
  Precision digunakan untuk mengukur ketepatan model dalam memprediksi data serangan.

$$Precision = \frac{TP}{TP+FP} \quad (5)$$

- *Recall*
  Recall digunakan untuk mengukur kemampuan *model* dalam mendeteksi seluruh data serangan.

$$Recall = \frac{TP}{TP+FN} \quad (6)$$

- *F1-score*
  F1-score digunakan untuk mengukur keseimbangan antara *precision* dan *recall* dalam proses klasifikasi.

$$F1 = 2\,x\,\frac{Precision\ x\ Recall}{Precision\ x\ Recall} \quad (7)$$

3. HASIL DAN PEMBAHASAN

Pada bagian ini dijelaskan hasil pengolahan serta analisis data *log ModSecurity* menggunakan metode *machine learning* untuk mengklasifikasikan *traffic web* normal dan serangan *SQL Injection*. Tahapan penelitian mencakup *preprocessing* data, *feature engineering* berbasis *TF-IDF*, pelatihan model, serta evaluasi performa menggunakan *accuracy, precision, recall, F1-score*, dan *confusion matrix* guna mengetahui kemampuan model dalam mendeteksi serangan berdasarkan karakteristik *request HTTP* pada *log server*.

Nuroji: * Penulis Korespondensi

Journal of Artificial Intelligence and Technology Information
***Website* : https://ejournal.techcart-press.com/index.php/jaiti**
E-ISSN 2985-6396
P-ISSN 2985-5306
Volume 4, Number 2, Juni 2026
https://doi.org/10.58602/jaiti.v4i2.262
Page 287-299

Data Collection

Pada tahap pengumpulan data (*data collection*) dalam penelitian ini, peneliti menerapkan *Web Application Firewall* (WAF) ModSecurity di perangkat webserver. Penerapan ini menghasilkan log audit (*modsec_audit.log*) dalam format JSON. Log tersebut memuat berbagai informasi penting terkait aktivitas serangan, seperti alamat IP penyerang, metode permintaan (*request method*), URI, *payload*, serta status respons HTTP yang merepresentasikan aktivitas dari permintaan tersebut.

Peneliti mengambil data berupa *file modsec_audit.log*, kemudian melakukan proses analisis deteksi serangan pada aplikasi web seperti yang ditunjukkan pada alur penelitian Gambar 1. Sementara itu, isi dari *file modsec_audit.log* yang berformat JSON ditampilkan pada Gambar 2.

Preprocessing

Pada tahap *preprocessing*, dilakukan proses parsing terhadap *log ModSecurity* untuk mengambil atribut-atribut penting yang akan digunakan dalam proses klasifikasi data. Hasil *parsing* menghasilkan beberapa fitur seperti *request*_line*, modsec_message, country, status,* dan *content_length*.


```
"response": {
      "protocol": "HTTP/1.1",
      "status": 403,
      "headers": {
         "Strict-Transport-Security": "max-age=63072000; includeSubDomains; preload",
         "X-Frame-Options": "SAMEORIGIN",
         "X-Content-Type-Options": "nosniff",
         "Referrer-Policy": "no-referrer-when-downgrade",
         "Content-Length": "199",
         "Keep-Alive": "timeout=5, max=100",
         "Connection": "Keep-Alive",
         "Content-Type": "text/html; charset=iso-8859-1"
      },
      "body": "<!DOCTYPE HTML PUBLIC \"-//IETF//DTD HTML 2.0//EN\">\n<html><head>\n<title>403
Forbidden</title>\n</head><body>\n<h1>Forbidden</h1>\n<p>You don't have permission to access
```


Gambar 2. Data *log ModSecurity*

Pada Gambar 2 ditampilkan contoh isi file modsec_audit.log yang digunakan pada penelitian ini. File log tersebut memiliki format JSON (JavaScript Object Notation), yaitu format pertukaran data terstruktur yang umum digunakan dalam aplikasi web dan sistem keamanan jaringan. Format JSON dipilih karena memiliki struktur hierarki yang mudah dibaca oleh manusia maupun diproses secara otomatis oleh sistem atau program analisis data.

Tabel 1. Hasil Proses parsing

request_line	modsec_message	country	status	content_length
POST /wp-login.php HTTP/1.1	SQL Injection Attack Detected	US	403	86
POST /wp-login.php HTTP/1.1	SQL Injection Attack Detected	US	403	85
POST /wp-login.php HTTP/1.1	SQL Injection Attack Detected	US	403	93
POST /wp-login.php HTTP/1.1	SQL Injection Attack Detected	US	403	91
GET / HTTP/1.0			400	0
POST /wp-login.php HTTP/1.1	SQL Injection Attack Detected	GB	403	86
POST /wp-login.php HTTP/1.1	SQL Injection Attack Detected	GB	403	85
POST /wp-login.php HTTP/1.1	SQL Injection Attack Detected	GB	403	93

Nuroji: * Penulis Korespondensi

Journal of Artificial Intelligence and Technology Information
***Website* : https://ejournal.techcart-press.com/index.php/jaiti**
E-ISSN 2985-6396
P-ISSN 2985-5306
Volume 4, Number 2, Juni 2026
https://doi.org/10.58602/jaiti.v4i2.262
Page 287-299

POST /wp-login.php HTTP/1.1	SQL Injection Attack Detected	GB	403	91
GET /wp-admin/setup-config.php?step=1 HTTP/1.1		US	403	0

Tabel 1 menunjukkan hasil proses parsing terhadap file log ModSecurity yang sebelumnya masih berbentuk data mentah berformat JSON. Pada tahap ini, informasi penting dari log diekstraksi dan diubah menjadi data terstruktur berbentuk tabel sehingga lebih mudah dianalisis dan digunakan pada proses machine learning.

Hasil parsing menghasilkan beberapa fitur utama yang dianggap relevan dalam proses identifikasi aktivitas normal maupun serangan pada aplikasi web. Fitur tersebut meliputi *request_line, modsec_message, country,* status, *dan content_length*.

Feature Engineering Extraction

Tahap feature engineering dilakukan untuk mengekstraksi fitur penting dari data log hasil parsing agar dapat digunakan dalam proses klasifikasi machine learning. Beberapa fitur numerik dan biner dibentuk berdasarkan karakteristik payload dan pola serangan pada request HTTP.

Fitur yang digunakan meliputi payload_length, special_char_count, slash_count, dan dot_count untuk merepresentasikan karakteristik payload. Selain itu, dilakukan deteksi keyword serangan menggunakan fitur has_sql_keyword, has_rce_keyword, dan has_xss_keyword untuk mengidentifikasi indikasi serangan *SQL Injection, Remote Code Execution (RCE),* dan *Cross-Site Scripting (XSS)*.

Hasil feature engineering digunakan sebagai input pada tahap vectorization dan pelatihan model machine learning. Hasil feature engineering ditunjukkan pada Tabel 2 berikut.

Tabel 2. *Feature Engineering Extraction*

No	payload_length	special_char_count	slash_count	dot_count	has_sql_keyword	has_rce_keyword	has_xss_keyword
0	86	9	0	0	1	0	0
1	85	9	0	0	1	0	0
2	93	9	0	0	1	0	0
3	91	9	0	0	1	0	0
4	0	0	0	0	0	0	0
5	86	9	0	0	1	0	0
6	85	9	0	0	1	0	0
7	93	9	0	0	1	0	0
8	91	9	0	0	1	0	0
9	0	0	0	0	0	0	0

Labeling

Tahap labeling dilakukan untuk mengelompokkan data log ke dalam empat kelas, yaitu SQL Injection (SQLi), Normal, Local File Inclusion (LFI), dan Remote Code Execution

Nuroji: * Penulis Korespondensi

Journal of Artificial Intelligence and Technology Information
***Website* : https://ejournal.techcart-press.com/index.php/jaiti**
E-ISSN 2985-6396
P-ISSN 2985-5306
Volume 4, Number 2, Juni 2026
https://doi.org/10.58602/jaiti.v4i2.262
Page 287-299

(RCE). Hasil labeling menunjukkan distribusi dataset sebagai berikut: SQLi sebanyak 2362 data, Normal sebanyak 1049 data, LFI sebanyak 48 data, dan RCE sebanyak 8 data.

Distribusi data tersebut menunjukkan kondisi yang sangat tidak seimbang (imbalanced dataset), terutama pada kelas LFI dan RCE yang jumlahnya jauh lebih sedikit dibanding kelas lainnya. Oleh karena itu, ketidakseimbangan data ini perlu ditangani pada tahap preprocessing menggunakan teknik SMOTE (Synthetic Minority Over-sampling Technique) agar model memiliki performa klasifikasi yang lebih optimal dan tidak bias terhadap kelas mayoritas.

Tabel 3. *Labeling*

No	payload_length	special_char_count	slash_count	dot_count	has_sql_keyword	has_rce_keyword	has_xss_keyword	label
0	86	9	0	0	1	0	0	**SQLi**
1	85	9	0	0	1	0	0	**SQLi**
2	93	9	0	0	1	0	0	**SQLi**
3	91	9	0	0	1	0	0	**SQLi**
4	0	0	0	0	0	0	0	**Normal**
5	86	9	0	0	1	0	0	**SQLi**
6	85	9	0	0	1	0	0	**SQLi**
7	93	9	0	0	1	0	0	**SQLi**
8	91	9	0	0	1	0	0	**SQLi**
9	0	0	0	0	0	0	0	**Normal**

Train-Test Split

Tahap train-test split dilakukan untuk membagi dataset menjadi data latih (training) dan data uji (testing) dengan rasio 80:20. Sebanyak 80% data (2773 sampel) digunakan untuk pelatihan model, sedangkan 20% data (694 sampel) digunakan untuk pengujian. Proses pembagian dilakukan menggunakan teknik stratify untuk menjaga proporsi setiap kelas tetap seimbang pada data training dan testing sesuai distribusi awal dataset.

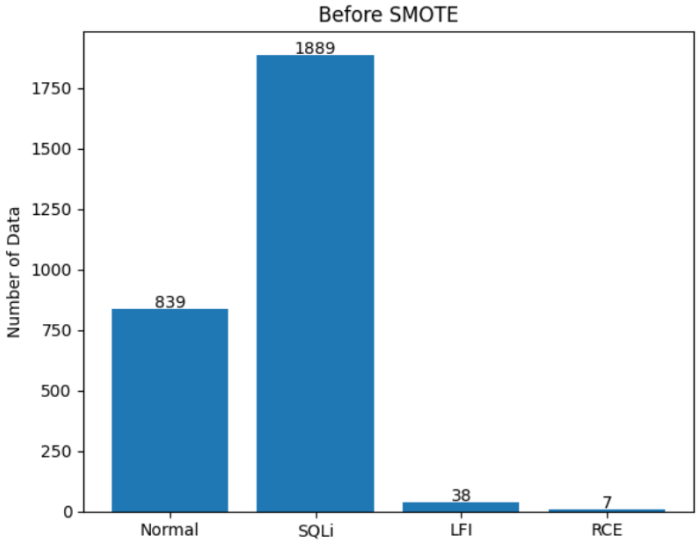
SMOTE Oversampling

Berdasarkan distribusi data sebelum penerapan SMOTE, terlihat bahwa dataset masih sangat tidak seimbang, dengan kelas SQLi sebanyak 1889 data, Normal 839 data, LFI 38 data, dan RCE hanya 7 data. Ketidakseimbangan ini berpotensi menyebabkan model

Nuroji: * Penulis Korespondensi

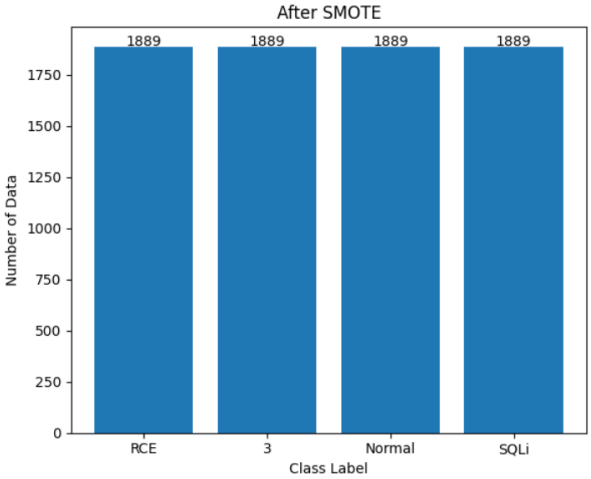
Journal of Artificial Intelligence and Technology Information
***Website* : https://ejournal.techcart-press.com/index.php/jaiti**
E-ISSN 2985-6396
P-ISSN 2985-5306
Volume 4, Number 2, Juni 2026
https://doi.org/10.58602/jaiti.v4i2.262
Page 287-299

menjadi bias terhadap kelas mayoritas dan kurang mampu mengenali kelas minoritas. Distribusi data sebelum SMOTE ditunjukkan pada Gambar 3.



Gambar 3. Sebelum dilakukan *SMOTE*

Untuk mengatasi masalah tersebut, diterapkan teknik SMOTE (Synthetic Minority Over-sampling Technique) pada data training. Setelah proses SMOTE, jumlah data pada setiap kelas menjadi seimbang, yaitu masing-masing sebanyak 1889 data untuk kelas SQLi, Normal, LFI, dan RCE. Dengan kondisi data yang seimbang, model diharapkan mampu mempelajari pola setiap kelas secara lebih adil sehingga meningkatkan performa klasifikasi, sebagaimana ditunjukkan pada Gambar 4.



Gambar 4. Setelah dilakukan *SMOTE*

Evaluation Machine learning Modeling

Pada tahap pemodelan, algoritma Support Vector Machine (SVM) diterapkan untuk mengklasifikasikan traffic web berdasarkan fitur hasil ekstraksi TF-IDF yang dikombinasikan dengan fitur numerik. Proses pelatihan model dilakukan menggunakan 2773 data training, sedangkan pengujian dilakukan pada 694 data testing untuk mengevaluasi performa model.

Nuroji: * Penulis Korespondensi

Journal of Artificial Intelligence and Technology Information
***Website* : https://ejournal.techcart-press.com/index.php/jaiti**
E-ISSN 2985-6396
P-ISSN 2985-5306
Volume 4, Number 2, Juni 2026
https://doi.org/10.58602/jaiti.v4i2.262
Page 287-299

Berdasarkan hasil pengujian, model SVM memperoleh accuracy sebesar 0.9812 (98.12%). Hasil classification report menunjukkan bahwa model memiliki performa sangat baik pada kelas mayoritas, khususnya kelas dengan support terbesar (kelas 3) dengan precision 1.00, recall 0.99, dan F1-score 1.00, serta kelas 1 dengan precision 0.98, recall 0.96, dan F1-score 0.97.

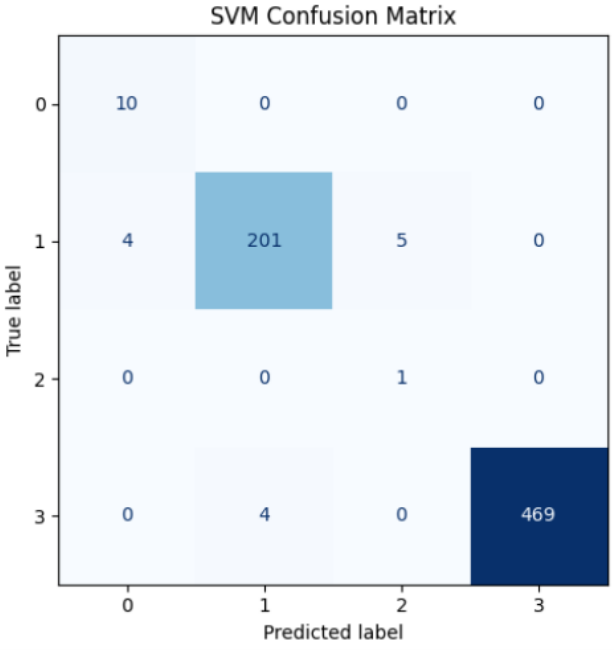
```
=== SVM RESULTS ===
Accuracy: 0.9812680115273775
              precision    recall  f1-score   support

           0       0.71      1.00      0.83        10
           1       0.98      0.96      0.97       210
           2       0.17      1.00      0.29         1
           3       1.00      0.99      1.00       473

    accuracy                           0.98       694
   macro avg       0.72      0.99      0.77       694
weighted avg       0.99      0.98      0.98       694
```

Gambar 5. Hasil pengujian *model SVM*

Confusion Matrix

Berdasarkan hasil confusion matrix SVM, model berhasil mengklasifikasikan sebagian besar data dengan baik, terutama pada kelas mayoritas yaitu kelas 1 (201 data benar) dan kelas 3 (469 data benar), meskipun masih terdapat beberapa kesalahan klasifikasi. Pada kelas 0 seluruh data (10 data) berhasil diklasifikasikan dengan benar, sedangkan pada kelas 2 hanya 1 data yang terdeteksi dengan benar.

Secara keseluruhan, model SVM memiliki performa yang baik, namun masih kurang optimal pada kelas dengan jumlah data kecil. Hal ini menunjukkan bahwa ketidakseimbangan dataset masih mempengaruhi kemampuan model dalam mengenali kelas minoritas.



Gambar 6. *Confusion Matrix Model SVM*

Nuroji: * Penulis Korespondensi

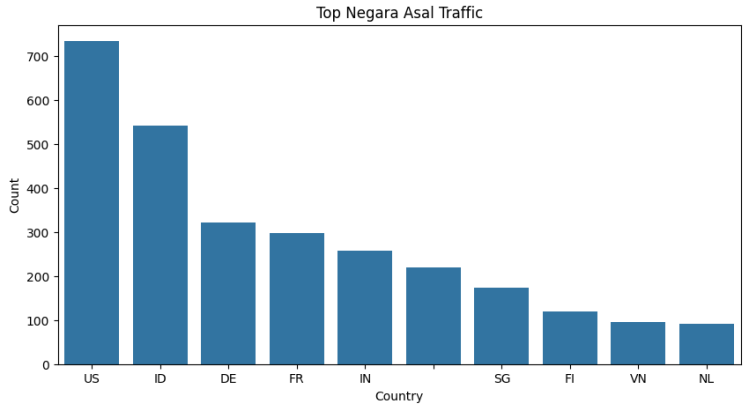
Journal of Artificial Intelligence and Technology Information
***Website* : https://ejournal.techcart-press.com/index.php/jaiti**
E-ISSN 2985-6396
P-ISSN 2985-5306
Volume 4, Number 2, Juni 2026
https://doi.org/10.58602/jaiti.v4i2.262
Page 287-299

Exploratory Data Analysis (EDA)

Tahap *Exploratory Data Analysis (EDA)* dilakukan untuk memahami karakteristik *dataset* sebelum proses pemodelan *machine learning*. Melalui *EDA*, peneliti dapat melihat distribusi data, asal *traffic* berdasarkan negara, serta jenis serangan yang dominan.

Top Negara Asal *Traffic*

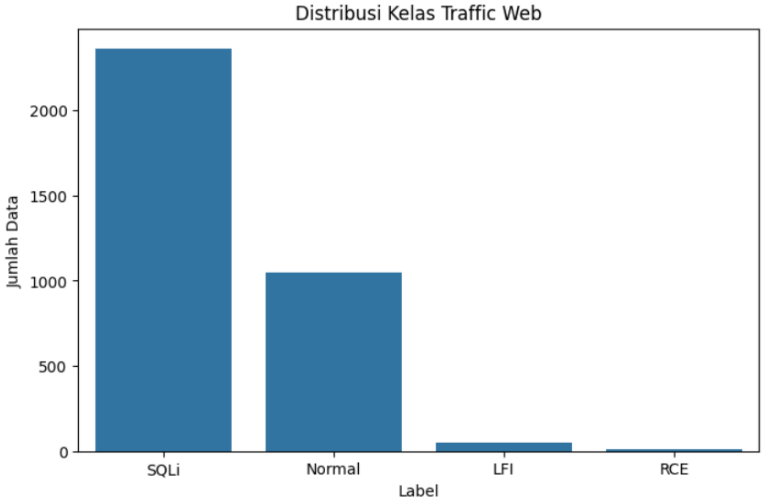
Berdasarkan visualisasi negara asal *traffic*, diketahui bahwa sebagian besar *request* berasal dari negara US, diikuti oleh ID, DE, FR, IN, dan beberapa negara lainnya seperti SG, FI, VN, dan NL. Tingginya *traffic* dari beberapa negara menunjukkan bahwa aktivitas *request* dan percobaan serangan terhadap *web server* berasal dari berbagai lokasi geografis.



Gambar 7. *Top Negara Asal Traffic*

Distribusi Jenis Serangan

Hasil visualisasi distribusi jenis serangan menunjukkan bahwa dataset didominasi oleh SQL Injection (SQLi) dengan jumlah 2362 data, diikuti oleh traffic Normal sebanyak 1049 data. Selain itu, terdapat kelas Local File Inclusion (LFI) sebanyak 48 data dan Remote Code Execution (RCE) sebanyak 8 data. Kondisi ini menunjukkan bahwa dataset memiliki ketidakseimbangan yang cukup signifikan antar kelas, terutama pada kelas LFI dan RCE yang jumlahnya sangat sedikit dibandingkan kelas lainnya.



Gambar 8. *Distribusi Jenis Serangan*

Nuroji: * Penulis Korespondensi

Journal of Artificial Intelligence and Technology Information
***Website* : https://ejournal.techcart-press.com/index.php/jaiti**
E-ISSN 2985-6396
P-ISSN 2985-5306
Volume 4, Number 2, Juni 2026
https://doi.org/10.58602/jaiti.v4i2.262
Page 287-299

4. KESIMPULAN

Penelitian ini menunjukkan bahwa algoritma Support Vector Machine (SVM) mampu mengklasifikasikan traffic web dengan baik berdasarkan log ModSecurity, dengan akurasi sebesar 98,12%. Sebelum penerapan SMOTE, dataset mengalami ketidakseimbangan signifikan pada kelas SQLi (1889), Normal (839), LFI (38), dan RCE (7). Setelah SMOTE, seluruh kelas pada data training menjadi seimbang masing-masing sebanyak 1889 data, sehingga model dapat belajar lebih adil terhadap setiap kelas. Hasil evaluasi menunjukkan performa sangat baik pada sebagian besar kelas, terutama kelas mayoritas, namun masih terdapat ketidakstabilan pada kelas dengan jumlah data sangat kecil. Secara keseluruhan, SVM dengan SMOTE efektif dalam mendeteksi serangan web, meskipun peningkatan pada kelas minoritas masih diperlukan.

5. REFERENCES

[1] W. Wahdana and K. H. Hanif, "Implementasi Keamanan Informasi Menggunakan Metode Web Application Firewall terhadap Serangan SQL Injection," *Jurnal Informatika Polinema*, vol. 11, no. 4, pp. 399–406, Aug. 2025, doi: 10.33795/jip.v11i4.7376.

[2] I. Darmawan, A. Nuridwan, A. Rahmatulloh, R. Gunawan, and R. Rizal, "Real-time Web Application Firewall Monitoring uses the OWASP CRS Framework," in *2024 Ninth International Conference on Informatics and Computing (ICIC)*, IEEE, Oct. 2024, pp. 1–6. doi: 10.1109/ICIC64337.2024.10956835.

[3] H. A. Gouda, M. A. Ahmed, and M. I. Roushdy, "Optimizing anomaly-based attack detection using classification machine learning," *Neural Comput Appl*, vol. 36, no. 6, pp. 3239–3257, Feb. 2024, doi: 10.1007/s00521-023-09309-y.

[4] Arif wicahyanto, N. Nurchim, and W. Wijiyanto, "PENERAPAN ARTIFICIAL NEURAL NETWORK DALAM DETEKSI SERANGAN PADA WEB SERVER APACHE," *Jurnal Informatika dan Rekayasa Elektronik*, vol. 8, no. 1, pp. 31–39, Apr. 2025, doi: 10.36595/jire.v8i1.1386.

[5] F. Ramadhan, I. Ruslianto, and S. Bahri, "Klasifikasi Serangan SQL Injection Menggunakan Algoritma Support Vector Machine Pada HTTP Request," *Coding: Jurnal Komputer dan Aplikasi*, vol. 13, no. 3, pp. 224–235, Dec. 2025, doi: 10.26418/coding.v13i3.92215.

[6] G. Indrawan, H. Setiawan, and A. Gunadi, "Multi-class SVM Classification Comparison for Health Service Satisfaction Survey Data in Bahasa," *HighTech and Innovation Journal*, vol. 3, no. 4, pp. 425–442, Dec. 2022, doi: 10.28991/HIJ-2022-03-04-05.

[7] C. Scano *et al.*, "ModSec-Learn: Boosting ModSecurity with Machine Learning," Jun. 2024, doi: 10.1007/978-3-031-76459-2_3.

[8] M. A. Elseddig and M. Mejri, "Incident Detection Based on Differential Analysis," *Journal of Information Security*, vol. 15, no. 03, pp. 378–409, 2024, doi: 10.4236/jis.2024.153022.

[9] M. A. Owaid and A. S. Hammoodi, "Evaluating Machine Learning and Deep Learning Models for Enhanced DDoS Attack Detection," *Mathematical Modelling of Engineering Problems*, vol. 11, no. 2, pp. 493–499, Feb. 2024, doi: 10.18280/mmep.110221.

[10] K. A. Cahyanto, M. A. al Hilmi, and M. Mustamiin, "Pengujian Rule-Based pada Dataset Log Server Menggunakan Support Vector Machine Berbasis Linear Discriminat Analysis untuk Deteksi Malicious Activity," *Jurnal Teknologi Informasi dan Ilmu Komputer*, vol. 9, no. 2, pp. 245–254, Feb. 2022, doi: 10.25126/jtiik.2022924107.

[11] A. Riverol, G. Betarte, R. Martínez, and Á. Pardo, "Capturing the security expert knowledge in feature selection for web application attack detection," Jul. 2024.

Nuroji: * Penulis Korespondensi

[12] A.-R. Al-Ghuwairi, Y. Sharrab, D. Al-Fraihat, M. AlElaimat, A. Alsarhan, and A. Algarni, "Intrusion detection in cloud computing based on time series anomalies utilizing machine learning," *Journal of Cloud Computing*, vol. 12, no. 1, p. 127, Aug. 2023, doi: 10.1186/s13677-023-00491-x.
[13] A. Shaheed and M. H. D. B. Kurdy, "Web Application Firewall Using Machine Learning and Features Engineering," *Security and Communication Networks*, vol. 2022, pp. 1–14, Jun. 2022, doi: 10.1155/2022/5280158.
[14] W. B. Demilie and F. G. Deriba, "Detection and prevention of SQLI attacks and developing compressive framework using machine learning and hybrid techniques," *J Big Data*, vol. 9, no. 1, p. 124, Dec. 2022, doi: 10.1186/s40537-022-00678-0
[15] M. Mujahid *et al.*, "Data oversampling and imbalanced datasets: an investigation of performance for machine learning and feature engineering," *J Big Data*, vol. 11, no. 1, p. 87, Jun. 2024, doi: 10.1186/s40537-024-00943-4.
[16] L. G. Cilento, P. S. G. de Mattos Neto, and D. C. Cunha, "A Framework for Efficient Pre-Processing of HTTP Requests Using Machine Learning-Based Web Application Firewalls," in *Anais do XXV Simpósio Brasileiro de Cibersegurança (SBSeg 2025)*, Sociedade Brasileira de Computação - SBC, Sep. 2025, pp. 18–31. doi: 10.5753/sbseg.2025.9793.
[17] Md. A. Talukder *et al.*, "A dependable hybrid machine learning model for network intrusion detection," *Journal of Information Security and Applications*, vol. 72, p. 103405, Feb. 2023, doi: 10.1016/j.jisa.2022.103405.
[18] S. Abbas *et al.*, "Artificial intelligence framework for heart disease classification from audio signals," *Sci Rep*, vol. 14, no. 1, p. 3123, Feb. 2024, doi: 10.1038/s41598-024-53778-7.
[19] R. F. Rahmat *et al.*, "Classifying Indonesian Cyber Crime Cases under ITE Law Using a Hybrid of Mutual Information and Support Vector Machine," *International Journal of Safety and Security Engineering*, vol. 13, no. 5, pp. 835–844, Nov. 2023, doi: 10.18280/ijsse.130507.
[20] H. C. Husada and A. S. Paramita, "Analisis Sentimen Pada Maskapai Penerbangan di Platform Twitter Menggunakan Algoritma Support Vector Machine (SVM)," *Teknika*, vol. 10, no. 1, pp. 18–26, Feb. 2021, doi: 10.34148/teknika.v10i1.311.
[21] N. I. Jabbar, "Support Vector Machine Prediction a Man in the Middle Attack on Traffic Networking," *Al-Nahrain Journal for Engineering Sciences*, vol. 28, no. 3, pp. 330–335, Sep. 2025, doi: 10.29194/NJES.28030330.
[22] A. BaniMustafa, M. Baklizi, and K. Khatatneh, "Machine Learning for Securing Traffic in Computer Networks," *International Journal of Advanced Computer Science and Applications*, vol. 13, no. 12, 2022, doi: 10.14569/IJACSA.2022.0131252.

Nuroji: * Penulis Korespondensi