JURNAL MEDIA INFORMATIKA [JUMIN]
Volume 7 Nomor 2 April 2026, Page 245-256
ISSN 2808-005X (media online)
Available Online at http://ejournal.sisfokomtek.org/index.php/jumin

Analisa Kerentanan Web Application Menggunakan Metode OWASP Top 10

Muhammad Reyhansyah Dwi Putra[1], Ari Winata[2], Muhammad Abel Fathir[3], Yoga Wahyu Prabowo[4], Sach Fathan Mulyana[5]

[1,2,3,4,5]Sistem Informasi, Universitas Bina Sarana Informatika, Jakarta Pusat, Indonesia

Email: [1]19232499@bsi.ac.id, [2]19232286@bsi.ac.id, [3]19231923@bsi.ac.id, [4]19230902@bsi.ac.id, [5] 19230779@bsi.ac.id
Email Penulis Korespondensi: [1]19232499@bsi.ac.id

Abstrak– Keamanan aplikasi web menjadi isu krusial seiring meningkatnya kompleksitas serangan siber, dengan kerentanan OWASP Top 10 sebagai vektor utama eksploitasi, sementara metode deteksi konvensional berbasis signature memiliki keterbatasan dalam menghadapi variasi serangan baru. Penelitian ini bertujuan mengembangkan sistem deteksi anomali keamanan web menggunakan algoritma XGBoost dengan pendekatan pattern recognition berbasis OWASP Top 10. Dataset CSIC HTTP 2010 dengan 61.065 sampel dibagi 80:20 untuk training dan testing. Feature engineering mengekstraksi 33 fitur teknis yang diseleksi menjadi 30 fitur terbaik menggunakan Mutual Information, meliputi analisis struktur URL, Shannon entropy, dan deteksi pola serangan OWASP. Model XGBoost dikonfigurasi dengan n_estimators=100, max_depth=8, learning_rate=0.1, dan cost-sensitive learning, dievaluasi menggunakan 5-fold cross-validation. Model mencapai akurasi 82,77%, precision 71,29%, recall 97,15%, F1-score 82,24%, dan ROC-AUC 93,29%. Pendekatan ini efektif meningkatkan deteksi anomali dengan recall tinggi dan berpotensi diimplementasikan sebagai lapisan proteksi tambahan pada Web Application Firewall.

Kata Kunci: XGBoost, keamanan aplikasi web, OWASP Top 10, deteksi anomali, analisis lalu lintas HTTP.

Abstract– Web application security has become critical as cyberattack complexity increases, with OWASP Top 10 vulnerabilities serving as primary exploitation vectors, while conventional signature-based detection methods have limitations in handling novel attack variations. This study aims to develop a web security anomaly detection system using the XGBoost algorithm with an OWASP Top 10-based pattern recognition approach. The CSIC HTTP 2010 dataset consisting of 61,065 samples was split 80:20 for training and testing. Feature engineering extracted 33 technical features reduced to 30 optimal features using Mutual Information, including URL structure analysis, Shannon entropy, and OWASP attack pattern detection. The XGBoost model was configured with n_estimators=100, max_depth=8, learning_rate=0.1, and cost-sensitive learning, evaluated using 5-fold cross-validation. The model achieved 82.77% accuracy, 71.29% precision, 97.15% recall, 82.24% F1-score, and 93.29% ROC-AUC. This approach effectively enhances anomaly detection with superior recall and shows strong potential for deployment as an additional protection layer in Web Application Firewall systems.

Keywords: XGBoost, web application security, OWASP Top 10, anomaly detection, HTTP traffic analysis.

1. PENDAHULUAN

Perkembangan teknologi informasi telah mendorong transformasi digital di berbagai sektor kehidupan. Aplikasi *web* kini menjadi infrastruktur vital dalam operasional bisnis, layanan publik, pendidikan, hingga interaksi sosial masyarakat. Seiring dengan meningkatnya ketergantungan terhadap aplikasi *web*, ancaman keamanan siber juga mengalami eskalasi yang signifikan[1]. Serangan terhadap aplikasi web menjadi semakin canggih dengan teknik yang terus berkembang, mulai dari serangan injeksi, skrip lintas situs, hingga eksekusi kode jarak jauh yang dapat mengakibatkan kerugian finansial dan reputasi organisasi.

Open Web Application Security Project (*OWASP*) secara konsisten merilis daftar periodik tentang risiko keamanan paling kritis dari perspektif keamanan aplikasi web. Kategori serangan seperti injeksi *SQL*, skrip lintas situs, dan kerentanan injeksi tetap menjadi ancaman persisten, dengan kerentanan injeksi berada di peringkat ketiga *OWASP Top 10* tahun 2021[2]. Kompleksitas serangan modern yang menggabungkan berbagai vektor serangan dan teknik penyamaran mempersulit deteksi menggunakan pendekatan berbasis tanda tangan tradisional.

Metode konvensional dalam keamanan *web* seperti *firewall* aplikasi web yang berbasis pencocokan pola aturan memiliki keterbatasan signifikan[3]. Sistem tersebut cenderung menghasilkan hasil positif palsu yang tinggi, kesulitan mendeteksi variasi serangan baru, dan memerlukan pembaruan aturan manual secara berkala[4]. Dalam konteks ini, pendekatan pembelajaran mesin menawarkan solusi adaptif yang mampu mempelajari pola serangan dari data *historis*[5].

XGBoost merupakan algoritma pembelajaran gabungan berbasis pohon keputusan yang menggunakan fitur peredaman statistik bertahap untuk mendeteksi pola berbahaya[6]. Pendekatan pembelajaran transfer dengan representasi *fitur multimodal* juga menunjukkan peningkatan signifikan dalam deteksi intrusi, dengan kemampuan menangani volume data besar dan mencapai akurasi hingga 98,2%[7]. Kombinasi *XGBoost* dengan rekayasa fitur berbasis pengenalan pola *OWASP* memungkinkan sistem untuk mengidentifikasi anomali keamanan dengan presisi tinggi[6].

This is an open access article under the CC–BY-SA license

Submitted: **23/12/2025**; Accepted: **30/01/2025**; Published: **02/04/2026**

JURNAL MEDIA INFORMATIKA [JUMIN]
Volume 7 Nomor 2 April 2026, Page 245-256
ISSN 2808-005X (media online)
Available Online at http://ejournal.sisfokomtek.org/index.php/jumin

Dataset CSIC HTTP 2010 dikembangkan oleh Dewan Riset Nasional Spanyol, yang berisi lalu lintas yang ditargetkan ke aplikasi *web* perdagangan elektronik [8]. *Dataset* ini terdiri dari 36.000 permintaan normal dan lebih dari 25.065 permintaan anomali, dengan permintaan *HTTP* yang dilabeli sebagai normal atau anomali dan mencakup serangan seperti injeksi *SQL*, kebocoran data, injeksi *CRLF*, skrip lintas situs, dan manipulasi parameter [8]. *Dataset* ini telah berhasil digunakan untuk deteksi keamanan web dalam penelitian sebelumnya. Ekstraksi fitur dari komponen permintaan *HTTP* seperti struktur *URL*, tajuk, muatan, dan *metadata* memungkinkan model untuk memahami karakteristik intrinsik dari setiap kategori serangan.

Meskipun berbagai penelitian telah mengeksplorasi penerapan *machine learning* untuk deteksi serangan *web*, beberapa keterbatasan masih ditemukan dalam literatur. Penelitian Ahmad et al.[9] menggunakan deep *learning* untuk *network intrusion detection* dan mencapai akurasi tinggi, namun memiliki *recall* rendah (85-88%) yang menyebabkan banyak serangan tidak terdeteksi, terutama pada kelas minoritas. Studi Dawadi et al[3]. mengimplementasikan deep *learning-enabled WAF* dengan kompleksitas komputasi tinggi yang kurang efisien untuk *real-time detection* pada *high-traffic environments.* Sementara itu, pendekatan berbasis signature yang diusulkan Khraisat et al.[4] gagal mendeteksi variasi serangan baru dan memerlukan pembaruan *rule* manual secara berkala. Penelitian Ullah et al[7]. menggunakan *ensemble learning* untuk *SQL injection detection*[10], namun hanya fokus pada satu jenis serangan tanpa pendekatan komprehensif multi-kategori *OWASP*. Lebih lanjut, sebagian besar penelitian terdahulu belum mengintegrasikan *pattern recognition* berbasis *OWASP* dengan *cost-sensitive learning* untuk menangani *class imbalance*, serta kurang memberikan interpretabilitas model dalam mengidentifikasi pola serangan spesifik.

Untuk mengatasi keterbatasan tersebut, penelitian ini mengusulkan pendekatan inovatif yang mengintegrasikan algoritma *XGBoost* dengan pattern detection engine berbasis *OWASP Top 10*. Novelty penelitian ini terletak pada tiga aspek utama. Pertama, pengembangan *systematic feature engineering* yang mengekstraksi 33 fitur teknis mencakup *URL structure analysis*, Shannon entropy measurement, *OWASP-based regex pattern detection*, dan *aggregated attack scores* yang memberikan representasi *komprehensif* dari karakteristik serangan. Kedua, implementasi *cost-sensitive learning* melalui pengaturan *scale_pos_weight* untuk mengoptimalkan *recall rate* tanpa mengorbankan *precision* secara signifikan, sehingga mengatasi masalah *class imbalance* yang umum terjadi pada dataset keamanan. Ketiga, kombinasi pattern recognition berbasis *OWASP* dengan *XGBoost* menghasilkan model yang tidak hanya akurat tetapi juga interpretable, memungkinkan *security analyst memahami feature importance* dan pola serangan yang terdeteksi untuk perbaikan *security policies.*

Berdasarkan *gap* dan *novelty* yang telah diidentifikasi, penelitian ini bertujuan untuk mengembangkan dan mengevaluasi sistem deteksi anomali keamanan aplikasi web menggunakan algoritma *XGBoost* dengan pendekatan *pattern recognition* berbasis *OWASP Top 10*. Secara spesifik, penelitian ini berfokus pada deteksi tujuh kategori kerentanan kritis, yaitu *SQL Injection, Cross-Site Scripting, Server-Side Template Injection, Command Injection, Remote Code Execution, Open Redirect*, dan *Insecure Direct Object Reference* (*IDOR*). Sistem mendeteksi lima kategori utama (*SQL Injection, XSS, SSTI, Command Injection*, dan *RCE*) yang memiliki sampel eksplisit dalam *dataset*, serta dua kategori tambahan (*Open Redirect* dan *IDOR*) melalui pola anomali umum, meskipun *dataset CSIC HTTP 2010* tidak menyediakan contoh eksplisit untuk kedua kategori tersebut. Setiap kategori diidentifikasi melalui pola tanda tangan khas menggunakan ekspresi reguler dan fitur statistik yang diekstraksi secara sistematis. *Dataset CSIC HTTP 2010* digunakan sebagai fondasi eksperimen dengan total 61.065 sampel *HTTP request*.

2. METODOLOGI PENELITIAN

2.1. Desain Penelitian

Penelitian ini menggunakan desain eksperimental kuantitatif dengan pendekatan *supervised learning* untuk klasifikasi biner (normal vs anomali) pada *lalu* lintas *HTTP*. Eksperimen dilakukan untuk membandingkan performa model *XGBoost* dengan baseline model dan mengevaluasi efektivitas *feature engineering* berbasis *OWASP pattern recognition* dalam meningkatkan kemampuan deteksi anomali keamanan *web*.

2.2. Populasi dan Sampel

Populasi penelitian adalah seluruh lalu lintas *HTTP* yang ditujukan ke aplikasi *web* e-commerce yang tercakup dalam dataset *CSIC HTTP 2010.* Sampel penelitian menggunakan teknik total sampling, di mana seluruh 61.065 *HTTP request* yang tersedia dalam *dataset* digunakan dalam eksperimen. *Dataset* dibagi menggunakan *stratified random sampling* dengan rasio 80:20 untuk *memastikan* proporsi kelas normal dan anomali tetap konsisten antara *training set* (48.852 sampel) dan *testing set* (12.213 sampel). *Stratified* sampling dipilih untuk mengatasi *class imbalance* (rasio normal:anomali = 1.436:1) dan memastikan representasi yang adil dari kedua kelas dalam proses *training* dan *testing*.

2.3. Instrumen Penelitian

Penelitian ini menggunakan instrumen berupa perangkat keras dan perangkat lunak sebagai berikut:

a. Perangkat Keras
   1. Processor : Apple M1 (8-core CPU)

This is an open access article under the CC–BY-SA license

Terakreditasi SINTA 5 SK :72/E/KPT/2024
Submitted: **23/12/2025**; Accepted: **30/01/2025**; Published: **02/04/2026**

JURNAL MEDIA INFORMATIKA [JUMIN]
Volume 7 Nomor 2 April 2026, Page 245-256
ISSN 2808-005X (media online)
Available Online at http://ejournal.sisfokomtek.org/index.php/jumin

2. RAM : 8 GB
3. Storage : 256 GB SSD
4. GPU : Apple M1 Integrated GPU (8-core)

b. Perangkat Lunak
1. Sistem Operasi : macOS (64-bit, Apple Silicon M1)
2. Python : Python 3.10.13
3. Libraries : XGBoost versi 1.5.0; scikit-learn versi 1.0.2; pandas versi 1.3.5; numpy versi 1.21.5; matplotlib versi 3.5.1; dan seaborn versi 0.11.2.
4. Development Environment: Jupyter Notebook 7.4.7

c. Dataset
1. Nama : CSIC HTTP 2010
2. Sumber : Spanish Research National Council
3. URL : Dataset resmi CSIC; dataset diperoleh melalui repositori publik karena situs resmi tidak lagi dapat diakses.
4. Format : raw HTTP request
5. Lisensi : Publik domain untuk penelitian akademik

2.4. Alur Penelitian

Penelitian ini dilakukan dengan mengikuti tahapan *sistematis* yang tergambar pada Gambar 1. Berikut adalah penjelasan setiap tahapan:

a. Pengumpulan Dataset
Dataset CSIC HTTP 2010 diunduh dari repositori publik. *Dataset* berisi 61.065 *HTTP request* yang terdiri dari 36.000 *request* normal dan 25.065 *request* anomali. *Dataset* dipilih karena memiliki label yang jelas dan sering digunakan dalam penelitian serupa.

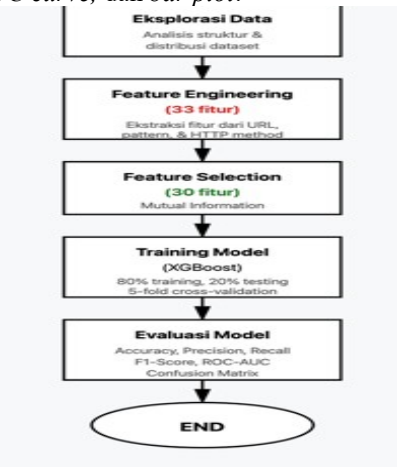
b. Eksplorasi Data
Data dimuat menggunakan *Python pandas* untuk pengecekan struktur, distribusi kelas, dan karakteristik *URL*. Tahap ini bertujuan memahami kondisi *dataset* sebelum diproses lebih lanjut.

c. Feature Engineering
Fitur yang diekstrak meliputi panjang URL, entropy, jumlah parameter, kedalaman path, karakter spesial, pattern OWASP, dan HTTP method dengan total 33 fitur.

d. Feature Selection
Menggunakan metode *Mutual Information*, dipilih 30 fitur terbaik untuk mengurangi dimensi data dan fokus pada fitur yang paling berpengaruh terhadap klasifikasi.

e. Training Model
Data dibagi menjadi 80% training dan 20% testing. Model *XGBoost* dilatih dengan cost-sensitive learning untuk menangani ketidakseimbangan kelas. Validasi dilakukan dengan *5-fold cross-validation*.

f. Evaluasi
Model dievaluasi menggunakan *metrik accuracy, precision, recall, F1-score,* dan *ROC-AUC*. Hasil divisualisasikan menggunakan *Confusion Matrix, ROC curve,* dan *bar plot*.



Gambar 1. Alur Penelitian

2.5 Dataset

This is an open access article under the CC–BY-SA license

Terakreditasi SINTA 5 SK :72/E/KPT/2024
Submitted: **23/12/2025**; Accepted: **30/01/2025**; Published: **02/04/2026**

JURNAL MEDIA INFORMATIKA [JUMIN]
Volume 7 Nomor 2 April 2026, Page 245-256
ISSN 2808-005X (media online)
Available Online at http://ejournal.sisfokomtek.org/index.php/jumin

Dataset yang digunakan adalah *CSIC HTTP 2010* dari *Spanish Research National Council* yang dipilih karena tersedia publik dan memiliki label valid untuk penelitian *web security*.

a. Karakteristik Dataset
Total data sebanyak 61.065 *HTTP* request terdiri dari 36.000 data normal (58.95%) dan 25.065 data anomali (41.05%). Jenis serangan meliputi *SQL Injection, XSS, Buffer Overflow*, *CRLF Injection*, *Path Traversal*, dan *Command Injection*. Dataset disimpan dalam format CSV dengan kolom *URL*, *Method*, dan *classification*. Rasio ketidakseimbangan kelas adalah 1.436:1 (dibulatkan menjadi 1.44:1). Karakteristik lengkap *dataset* ditampilkan pada Tabel 1.

Tabel 1. Karakteristik Dataset CSIC HTTP 2010

Karakteristik	Nilai
Total Request	61.065
Request Normal	36.000 (58.95%)
Request Anomali	25.065 (41.05%)
Rasio Imbalance	1.436:1 (normal:anomali)
Format File	CSV
Kolom Utama	URL, Method, classification
Jenis Serangan	SQLi, XSS, Buffer Overflow, CRLF, Path Traversal, Command Injection

b. Preprocessing
Dataset dipraproses melalui penghapusan kolom tidak relevan, normalisasi label *biner* (0 normal, 1 anomali), validasi nilai hilang, *URL decoding,* serta pembagian data menggunakan stratified sampling.

2.6 Prosedur Preprocessing

Prosedur pengumpulan data dilakukan melalui tahapan berikut:

a. URL Decoding
Semua URL yang ter-encode (mengandung karakter %XX) di-decode menggunakan fungsi urllib.parse.unquote() untuk mengembalikan karakter asli. Contoh: %3Cscript%3E menjadi <script>.
b. Normalisasi Label
Label klasifikasi dinormalisasi menjadi format biner: 0 untuk request normal dan 1 untuk request anomali.
c. Handling Missing Values
Pengecekan missing values dilakukan menggunakan pandas.isnull(). Hasil menunjukkan tidak ada missing values dalam dataset (0% missing).
d. Penghapusan Kolom Tidak Relevan
Kolom yang tidak berkontribusi terhadap klasifikasi seperti timestamp dan session ID dihapus dari dataset.
e. Data Splitting
Dataset dibagi menggunakan stratified *train_test_split* dari *scikit-learn* dengan *parameter*:
   1. test_size = 0.2 (20% untuk testing)
   2. random_state = 42 (untuk reproducibility)
   3. stratify = y (mempertahankan proporsi kelas)

   Hasil splitting:
   1. Training set: 48.852 samples (36.000 × 0.8 normal + 25.065 × 0.8 anomali)
   2. Testing set: 12.213 samples (36.000 × 0.2 normal + 25.065 × 0.2 anomali)

2.7 Pattern Detection Engine

Pattern detection engine dibuat menggunakan *regular expression* untuk mendeteksi pola serangan *OWASP. Engine* terdiri dari 7 *modul detector* yang ditampilkan pada Tabel 2:

Tabel 2. Modul Pattern Detection Engine

Modul	Pattern yang Dideteksi	Contoh
SQL Injection	Keyword SQL, Operator, Comment, Quote	SELECT, UNION, OR, --, '
XSS	HTML Tag, Event Handler, Protocol	script, iframe, onload, javascript:

This is an open access article under the CC–BY-SA license

Terakreditasi SINTA 5 SK :72/E/KPT/2024
Submitted: **23/12/2025**; Accepted: **30/01/2025**; Published: **02/04/2026**

JURNAL MEDIA INFORMATIKA [JUMIN]
Volume 7 Nomor 2 April 2026, Page 245-256
ISSN 2808-005X (media online)
Available Online at http://ejournal.sisfokomtek.org/index.php/jumin

SSTI	Template Delimiter, Magic Method	{{ }}, {% %}, class
Command Injection	Command Separator, Executable	; | & `, cat, bash
RCE	Dangerous Function	eval, exec, system
Open Redirect	Redirect Parameter + URL	redirect=http://, url=https://
IDOR	Parameter ID + Numeric	Id=123, user=667

a. SQL Injection Detector
Mendeteksi keyword SQL (*SELECT, UNION, INSERT, DELETE*), operator (*OR, AND*), *comment SQL* (*double dash, slash asterisk, hash*), dan *quote characters*. Detector ini mengidentifikasi pola-pola yang umum digunakan dalam serangan *SQL injection*, di mana penyerang mencoba memanipulasi *query database* melalui input yang tidak tersanitasi untuk mengakses atau memodifikasi data secara tidak sah.

b. XSS Detector
Mendeteksi *tag HTML* berbahaya (*script, iframe, img*), *event handler (onload, onerror)*, dan *protocol javascript. Cross-Site Scripting* (*XSS*) detector menangkap upaya injeksi kode *JavaScript* berbahaya yang dapat dieksekusi di browser korban, memungkinkan penyerang mencuri *session, cookies*, atau melakukan tindakan atas nama pengguna yang sah.

c. SSTI Detector
Mendeteksi *delimiter template (curly braces ganda, curly percent)* dan *Python magic method* seperti *class* dan mro. *Server-Side Template Injection (SSTI)* detector mengidentifikasi upaya eksploitasi *template engine* di sisi server yang dapat menyebabkan eksekusi kode *arbitrary* dan kompromi sistem secara keseluruhan.

d. Command Injection Detector
Mendeteksi separator *command* (*semicolon, pipe, ampersand, backtick, dollar*) dan *executable* (*cat, ls, wget, bash*). Detector ini menangkap karakter dan perintah yang digunakan untuk menjalankan *command* sistem operasi melalui aplikasi *web*, yang dapat memberikan akses *shell* kepada penyerang untuk mengontrol *server*.

e. RCE Detector
Mendeteksi fungsi berbahaya seperti *eval, exec, system, dan shell_exec. Remote Code Execution (RCE)* detector mengidentifikasi fungsi-fungsi yang memungkinkan eksekusi kode dinamis, yang merupakan vektor serangan kritis karena dapat memberikan kontrol penuh terhadap aplikasi dan *server*.

f. Open Redirect Detector
Mendeteksi *parameter redirect* yang mengarah ke *URL external*. Detector ini mengidentifikasi kerentanan di mana aplikasi dapat dimanipulasi untuk mengarahkan pengguna ke situs berbahaya, sering digunakan dalam serangan *phising* untuk mencuri kredensial dengan memanfaatkan kepercayaan terhadap domain asli.

g. IDOR Detector
Mendeteksi pola akses *direct object* melalui parameter *id numerik. Insecure Direct Object Reference (IDOR)* detector mengidentifikasi upaya akses tidak sah ke *resource* dengan memanipulasi *parameter identifier*, yang dapat menyebabkan kebocoran data sensitif milik pengguna lain tanpa otorisasi yang *proper*.

2.8 Feature Engineering

Proses ekstraksi fitur menghasilkan beberapa kategori fitur yang dirangkum dalam Tabel 3:

Tabel 3. Kategori Fitur yang Diekstrak

Kategori	Jumlah Fitur	Deskripsi
URL Structure	7	Panjang, entropy, parameter count, path depth, special chars, encoding.
Pattern Detection	13	Binary features dari OWASP pattern engine
Aggregated Scores	7	Total score per kategori serangan (SQLi, XSS, SSTI, CMD, RCE, IDOR, Redirect)
HTTP Method	5	One-hot encoding GET, POST, PUT, DELETE, Other
Statistical	1	Log transformation panjang URL
Total	33	Sebelum feature selection

This is an open access article under the CC–BY-SA license

Submitted: **23/12/2025**; Accepted: **30/01/2025**; Published: **02/04/2026**

JURNAL MEDIA INFORMATIKA [JUMIN]
Volume 7 Nomor 2 April 2026, Page 245-256
ISSN 2808-005X (media online)
Available Online at http://ejournal.sisfokomtek.org/index.php/jumin

Terpilih	30	Setelah Mutual Information selection

a. URL Structure Features
Fitur *URL Structure* diekstraksi langsung dari *string URL* untuk menangkap karakteristik statistik dan struktural yang umum muncul pada request berbahaya. Tujuh fitur digunakan, yaitu: panjang *URL*, *entropy* karakter, jumlah parameter, kedalaman *path*, jumlah karakter spesial, keberadaan *encoding*, dan rasio karakter numerik.
Panjang URL dihitung sebagai jumlah total karakter dalam string URL:

$$url_{length} = |URL| \quad (1)$$

Tingkat keacakan karakter URL diukur menggunakan Shannon entropy:

$$H(X) = -\sum_{i=1}^{n} \square P(x_i) P(x_i) \quad (2)$$

dimana $P(x_i)$merupakan probabilitas kemunculan karakter $x_i$dalam URL. Nilai entropy yang tinggi menunjukkan distribusi karakter yang lebih acak, yang umum ditemukan pada payload serangan yang di-encode atau disamarkan. Jumlah *parameter query* dan kedalaman path dihitung berdasarkan struktur *URL*, sedangkan jumlah karakter spesial dihitung dari kemunculan karakter seperti %, &, dan =. Keberadaan *URL encoding* direpresentasikan sebagai fitur biner, sementara rasio karakter numerik dihitung sebagai perbandingan jumlah digit terhadap total panjang *URL*.

b. Pattern Detection Features
Output dari *pattern engine* untuk setiap kategori *OWASP* menghasilkan *13 binary features*, Setiap detector menghasilkan nilai 0 atau 1 yang mengindikasikan keberadaan pola serangan spesifik, sehingga membentuk *signature* unik dari berbagai jenis ancaman yang terdeteksi dalam satu request.

c. Aggregated Scores
Aggregated scores mengakumulasi total kemunculan pola berbahaya per kategori serangan (*SQLi, XSS, SSTI, CMD, RCE, IDOR, Redirect)*, memberikan skor intensitas ancaman dalam satu *request*. Fitur ini merepresentasikan tingkat keparahan serangan, bukan hanya keberadaannya.

d. HTTP Method Features
Fitur ini merepresentasikan metode *HTTP* yang digunakan (*GET, POST, PUT, DELETE, Other*) melalui *one-hot encoding*. Karena jenis serangan tertentu cenderung menggunakan metode spesifik (misalnya, *SQLi* sering *via GET*), fitur ini membantu model membedakan profil risiko berdasarkan metode permintaan.

e. Statistical Features
Fitur utama adalah *url_length_log*, yaitu logaritma natural dari panjang *URL* yang digunakan untuk normalisasi distribusi panjang *URL* yang cenderung miring ke kanan. Transformasi logaritmik membantu model menangani *outlier* dan memberikan representasi yang lebih stabil untuk *URL* yang sangat panjang.

Total fitur yang dihasilkan adalah 33 fitur yang kemudian diseleksi menjadi 30 fitur terbaik menggunakan metode *Mutual Information* untuk menghilangkan fitur dengan *information gain* rendah dan meningkatkan efisiensi komputasi model[11].

2.9 Model XGBoost

Algoritma XGBoost. digunakan dengan konfigurasi parameter sebagai berikut: *n_estimators* sebanyak 100 untuk jumlah *decision tree, max_depth* bernilai 8 untuk kedalaman maksimum *tree, learning_rate* sebesar 0.1 untuk laju pembelajaran, *subsample* sebesar 0.8 untuk proporsi sampel per *tree, scale_pos_weight* dihitung otomatis untuk *cost-sensitive learning*, dan *random_state* bernilai 42 untuk *reproducibility*

a. Cost-Sensitive Learning
Cost-sensitive learning mengatasi ketidakseimbangan kelas dengan memberikan bobot lebih tinggi pada kelas minoritas (anomali). Parameter *scale_pos_weight* dihitung berdasarkan rasio jumlah sampel normal terhadap anomali, sehingga model lebih sensitif dalam mendeteksi kelas yang lebih sedikit jumlahnya. Pendekatan ini meningkatkan kemampuan deteksi anomali tanpa mengabaikan kelas mayoritas, dengan nilai *scale_pos_weight* sebesar 1.436 yang mencerminkan rasio ketidakseimbangan kelas 1.436:1 dalam *dataset*.

b. Training
Model dilatih menggunakan training set (80% data) dengan *5-fold cross-validation* untuk memastikan stabilitas performa. Proses training menggunakan pembagian data 80:20 untuk training dan testing, di mana training set dibagi lagi menjadi 5 lipatan (*fold*) untuk validasi silang. Setiap *fold* bergantian menjadi *validation set* sementara 4 *fold* lainnya untuk training, sehingga menghasilkan 5 model yang performanya dirata-ratakan. Metode ini memastikan model tidak *overfitting* dan memiliki performa yang konsisten pada data yang belum pernah dilihat sebelumnya.

2.10 Evaluasi

Evaluasi dilakukan pada test set (20% data) menggunakan metrik standar klasifikasi:

This is an open access article under the CC–BY-SA license

Terakreditasi SINTA 5 SK :72/E/KPT/2024
Submitted: **23/12/2025**; Accepted: **30/01/2025**; Published: **02/04/2026**

JURNAL MEDIA INFORMATIKA [JUMIN]
Volume 7 Nomor 2 April 2026, Page 245-256
ISSN 2808-005X (media online)
Available Online at http://ejournal.sisfokomtek.org/index.php/jumin

a. Accuracy

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (3)$$

b. Precision

$$Precision = \frac{TP}{TP + FP} \quad (4)$$

c. Recall

$$Recall = \frac{TP}{TP + FN} \quad (5)$$

d. F1-Score

$$F_1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (6)$$

e. ROC-AUC
Area under ROC curve (Receiver Operating Characteristic - Area Under Curve) untuk mengukur kemampuan model membedakan kelas. *ROC*-AUC mengevaluasi performa model di berbagai *threshold* klasifikasi dengan nilai 0-1, di mana nilai mendekati 1 menunjukkan model sangat baik dalam memisahkan kelas normal dan anomali tanpa bias terhadap *threshold* tertentu.

f. Confusion Matrix
Menganalisis distribusi prediksi: TN (normal diprediksi normal), FP (normal diprediksi anomali), FN (anomali diprediksi normal), dan TP (anomali diprediksi anomali). Metrik ini memberikan breakdown lengkap performa klasifikasi yang memungkinkan analisis error secara detail, di mana FP mengindikasikan *false alarm* yang dapat mengganggu operasional, sementara FN adalah *missed detection* yang lebih berbahaya karena ancaman lolos dari deteksi.

g. Visualisasi
Hasil divisualisasikan menggunakan *Confusion Matrix* dengan *seaborn*, *ROC curve*, *bar plot* untuk *cross-validation metrics,* dan *horizontal bar plot* untuk *feature importance*. Visualisasi multi-dimensi ini memudahkan interpretasi performa model secara *komprehensif*, mulai dari akurasi prediksi per kelas*, trade-off* antara true positive rate dan *false positive rate*, konsistensi performa *across folds*, hingga kontribusi relatif setiap fitur terhadap keputusan model.

3. HASIL DAN PEMBAHASAN

3.1 Hasil Rekayasa Fitur

Proses *feature engineering* menghasilkan 33 fitur yang merepresentasikan karakteristik struktural *URL*, pola serangan berbasis *OWASP*, serta sifat statistik lalu lintas *HTTP* pada *dataset CSIC HTTP 2010*. Analisis korelasi menunjukkan bahwa fitur yang berkaitan dengan keberadaan karakter ter-encode dan kompleksitas struktur *URL* memiliki hubungan paling kuat dengan permintaan anomali. Temuan ini mengindikasikan bahwa teknik penyamaran seperti *URL encoding* dan manipulasi karakter merupakan strategi umum yang digunakan penyerang untuk menyembunyikan *payload* berbahaya dari mekanisme validasi input dan *Web Application Firewall* berbasis aturan[3][4].

Fitur berbasis entropi dan struktur *URL* juga menunjukkan relevansi yang tinggi dalam membedakan lalu lintas berbahaya dan *legitimate*. Permintaan anomali cenderung memiliki URL yang lebih panjang, lebih kompleks, dan memiliki tingkat ketidakberaturan karakter yang lebih tinggi. Pola ini mencerminkan variasi *payload* yang lebih besar akibat penggunaan karakter acak, *encoding* berlapis, serta injeksi parameter, yang jarang ditemukan pada permintaan *HTTP* normal[5]. Secara konseptual, hasil ini sejalan dengan teori deteksi anomali berbasis statistik yang menyatakan bahwa serangan cenderung meningkatkan kompleksitas dan entropi input.

Selain karakteristik struktural, fitur deteksi pola yang berkaitan dengan *SQL Injection* menunjukkan kontribusi yang signifikan. Keberadaan karakter *quote* dan operator *SQL* berfungsi sebagai indikator penting karena merupakan elemen fundamental dalam eksploitasi *query SQL*[7][12]. Skor agregat dari berbagai pola *SQL Injection* terbukti lebih informatif dibandingkan deteksi pola individual, mengindikasikan bahwa kombinasi beberapa sinyal serangan memberikan *prediktor* yang lebih andal untuk identifikasi anomali keamanan *web*.

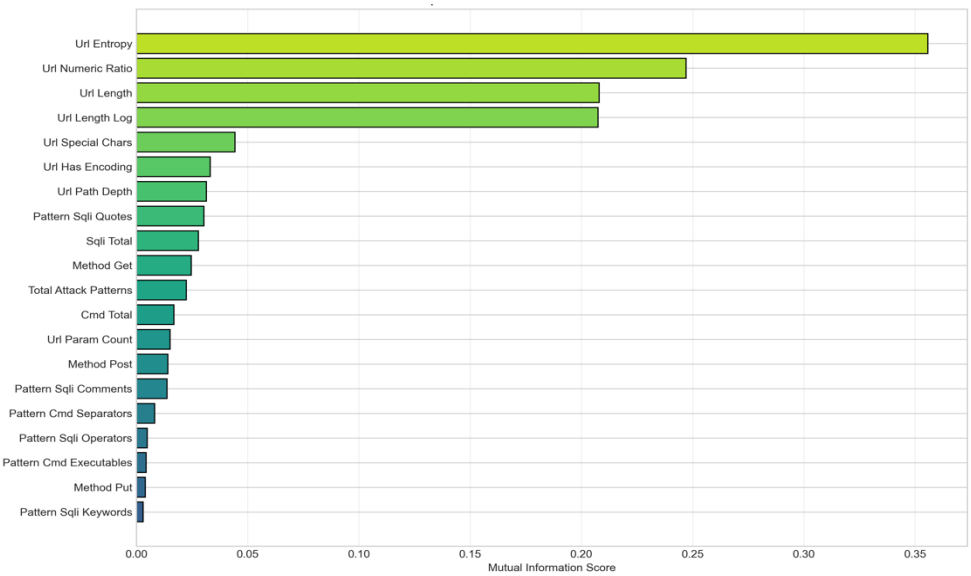
Analisis statistik deskriptif lebih lanjut memperkuat temuan tersebut dengan menunjukkan perbedaan distribusi yang konsisten antara permintaan normal dan anomali. Secara umum, permintaan anomali memiliki struktur URL yang lebih kompleks, jumlah parameter yang lebih banyak, serta akumulasi pola serangan yang lebih tinggi. Temuan ini menegaskan bahwa pendekatan yang mengombinasikan fitur struktural, statistik, dan berbasis domain knowledge OWASP mampu menangkap karakteristik fundamental serangan web secara efektif, sehingga layak dijadikan fondasi bagi sistem deteksi anomali keamanan aplikasi web.Dari 33 fitur yang diekstraksi, seleksi fitur menggunakan metode

This is an open access article under the CC–BY-SA license

Terakreditasi SINTA 5 SK :72/E/KPT/2024
Submitted: **23/12/2025**; Accepted: **30/01/2025**; Published: **02/04/2026**

JURNAL MEDIA INFORMATIKA [JUMIN]
Volume 7 Nomor 2 April 2026, Page 245-256
ISSN 2808-005X (media online)
Available Online at http://ejournal.sisfokomtek.org/index.php/jumin

Mutual Information berhasil mengidentifikasi 30 fitur terbaik yang memiliki *information* gain tertinggi terhadap variabel target.



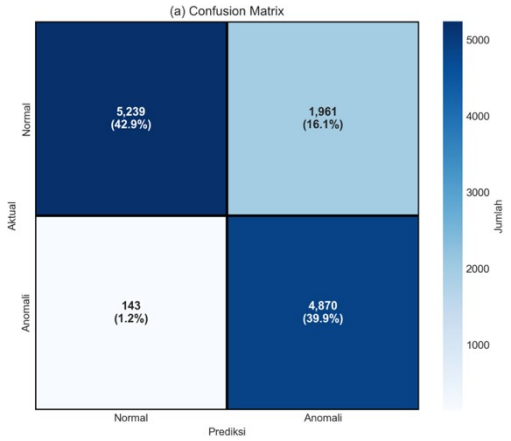
Gambar 2. Horizontal Bar Plot - Mutual Information Score

Gambar 2 menampilkan peringkat fitur berdasarkan skor *Mutual Information*. Fitur *url_entropy* menunjukkan *MI score* tertinggi (0.3558), mengkonfirmasi bahwa entropy merupakan prediktor terkuat untuk anomali dengan *information gain* tertinggi. Fitur *url_numeric_ratio* (0.2471) dan *url_length* (0.2080) menempati posisi kedua dan ketiga, menunjukkan bahwa karakteristik struktural *URL* sangat informatif untuk deteksi anomali. Transformasi logaritmik dari panjang *URL* yaitu *url_length_log* dengan *MI score* 0.2076 memberikan *information gain* yang hampir setara.

Fitur struktur *URL* mendominasi 7 peringkat teratas, diikuti oleh fitur deteksi pola seperti *pattern_sqli_quotes* (0.0304) dan skor agregat seperti *sqli_total* (0.0278). Fitur metode *HTTP* juga menunjukkan kontribusi signifikan dengan *method_GET* (0.0246) dan *method_POST* (0.0141) masuk dalam 15 besar. Proses seleksi ini berhasil mengeliminasi 3 fitur dengan *MI score* terendah yang memiliki *information gain* minimal. Distribusi fitur terpilih menunjukkan representasi seimbang dari semua kategori: struktur *URL* (7 fitur), deteksi pola (11 fitur), skor agregat (7 fitur), metode *HTTP* (4 fitur), dan fitur statistik (1 fitur).

3.2 Performa Model XGBoost

Model *XGBoost* yang dilatih dengan konfigurasi parameter optimal menunjukkan performa yang baik dalam mendeteksi anomali keamanan *web*. Evaluasi pada test set (20% dari total data atau sekitar 12.213 request) menghasilkan metrik-metrik sebagai berikut: *accuracy* sebesar 82,77%, *precision* 71,29%, *recall* 97,15%, *F1-score* 82,24%, dan *ROC-AUC* 93.29%.



Gambar 3. Confusion Matrix

Gambar 3 menampilkan *Confusion Matrix* yang menunjukkan distribusi prediksi model. Dari 7.200 *request* normal dalam *test* set, model berhasil mengklasifikasikan 5.239 sebagai *True Negative (TN)* dengan benar, namun terdapat 1.961 *False Positive (FP)*. Sementara dari 5.013 *request* anomali, model berhasil mendeteksi 4.870 sebagai *True Positive (TP)* dengan hanya 143 *False Negative (FN)*.

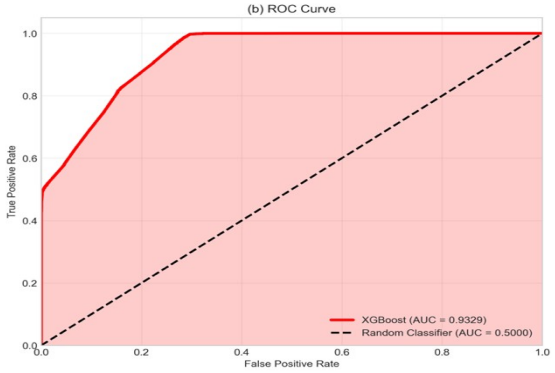
This is an open access article under the CC–BY-SA license

Submitted: **23/12/2025**; Accepted: **30/01/2025**; Published: **02/04/2026**

JURNAL MEDIA INFORMATIKA [JUMIN]
Volume 7 Nomor 2 April 2026, Page 245-256
ISSN 2808-005X (media online)
Available Online at http://ejournal.sisfokomtek.org/index.php/jumin

Nilai *recall* yang sangat tinggi (97,15%) menunjukkan bahwa model berhasil mendeteksi hampir semua serangan yang ada dalam *dataset*, dengan hanya 2.85% *false negative rate*. Hal ini sangat krusial dalam konteks keamanan siber[9], di mana *missed detection (false negative)* memiliki konsekuensi yang jauh lebih berbahaya dibandingkan *false alarm (false positive)*. Model mampu menangkap 97,15% dari seluruh serangan yang sebenarnya, memberikan tingkat proteksi yang tinggi terhadap ancaman *web security*[13].

Nilai *precision* sebesar 71,29% mengindikasikan bahwa dari semua *request* yang diprediksi sebagai anomali, sekitar 71% benar-benar merupakan serangan. Meskipun terdapat 27.24% *false positive rate, trade-off* ini dapat diterima mengingat pentingnya mendeteksi serangan. Dalam implementasi praktis, *false positive* dapat dimitigasi dengan *secondary validation* atau manual *review* oleh *security analyst*[3].



Gambar 4. ROC Curve dengan AUC Score 93.29%

Gambar 4 menampilkan kurva *ROC (Receiver Operating Characteristic)* yang menggambarkan *trade-off* antara *True Positive Rate* dan *False Positive Rate* di berbagai *threshold* klasifikasi. *ROC-AUC score* sebesar 93.29% menunjukkan kemampuan model yang sangat baik dalam membedakan antara kelas normal dan anomali. Kurva yang cenderung menuju sudut kiri atas menunjukkan bahwa model dapat mencapai *true positive rate* tinggi dengan *false positive rate* yang relatif rendah. Nilai ini mengindikasikan bahwa model memiliki *discriminative power* yang tinggi dan tidak bergantung pada *threshold* tertentu.

F1-score sebesar 82,24% merepresentasikan *harmonic mean* antara *precision* dan *recall*, memberikan gambaran seimbang tentang performa model. Nilai ini menunjukkan bahwa model berhasil mencapai keseimbangan yang baik antara kemampuan mendeteksi serangan (*recall*) dan akurasi prediksi positif (*precision*).

3.3 Validasi Silang dan Pentingnya Fitur

Hasil *5-fold cross-validation* menunjukkan konsistensi performa model dengan metrik yang sangat stabil. *Mean accuracy* sebesar 82.68% dengan standar deviasi sangat rendah (±0.49%) mengindikasikan bahwa model memiliki performa konsisten terlepas dari komposisi data latih. Variasi *accuracy* antar *fold* hanya berkisar 0.98 poin persentase (*range*: 82.19% - 83.17%). *Precision* rata-rata 71.25% (±0.59%) menunjukkan bahwa sekitar 71% dari prediksi positif model adalah benar, dengan range 70.66% - 71.84%.

Recall mencapai 96.95% dengan standar deviasi paling rendah (±0.42%), mengkonfirmasi bahwa kemampuan model dalam mendeteksi serangan sangat stabil dan andal. Di semua *fold*, model mampu menangkap minimal 96.53% dari seluruh serangan (*range*: 96.53% - 97.37%), yang merupakan pencapaian sangat baik dalam konteks sistem deteksi keamanan[9]. *F1-score* sebesar 82.13% (±0.44%) merepresentasikan *harmonic mean* yang seimbang antara *precision* dan *recall*, dengan *range* 81.69% - 82.57%. *ROC-AUC* mencapai 92.78% dengan standar deviasi ±0.90% (*range*: 91.88% - 93.68%), mengindikasikan kemampuan pemisahan kelas yang sangat baik dan konsisten.

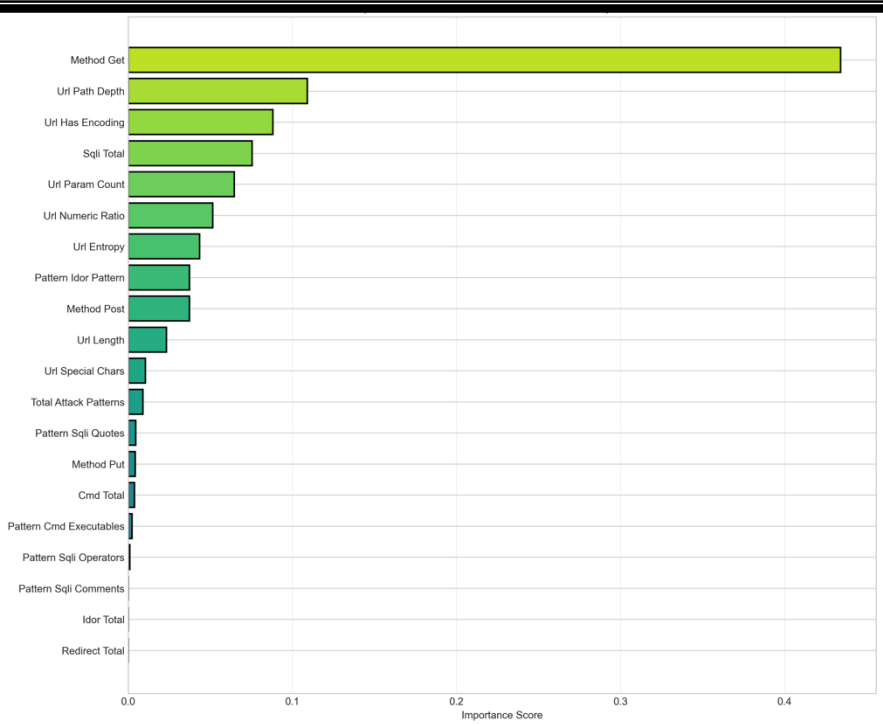
Standar deviasi yang sangat rendah di semua metrik (berkisar 0.42% - 0.90%) mengindikasikan bahwa model tidak *overfitting* dan memiliki kemampuan generalisasi yang sangat baik[14]. Variasi performa antar *fold* berada dalam rentang yang sangat ketat, membuktikan bahwa performa tinggi yang dicapai bukan hasil dari pembagian data yang beruntung, melainkan kemampuan intrinsik model dalam mempelajari pola serangan yang dapat digeneralisasi. Konsistensi hasil memvalidasi bahwa penyetelan *hyperparameter* yang dilakukan (n_estimators=100, max_depth=8, learning_rate=0.1) sudah optimal[11]. Waktu yang dibutuhkan untuk menjalankan *5-fold cross-validation* adalah 5.04 detik pada perangkat keras standar, menunjukkan bahwa model memiliki efisiensi komputasi yang baik.

Analisis *feature importance* menggunakan metrik gain dari *XGBoost* memberikan wawasan tentang kontribusi relatif setiap fitur terhadap pengambilan keputusan model.

This is an open access article under the CC–BY-SA license

Terakreditasi SINTA 5 SK :72/E/KPT/2024
Submitted: **23/12/2025**; Accepted: **30/01/2025**; Published: **02/04/2026**

JURNAL MEDIA INFORMATIKA [JUMIN]
Volume 7 Nomor 2 April 2026, Page 245-256
ISSN 2808-005X (media online)
Available Online at http://ejournal.sisfokomtek.org/index.php/jumin



Gambar 5. Horizontal Bar Plot - Top 20 Feature Importance

Gambar 5 menampilkan peringkat *feature importance* berdasarkan metrik gain dari *XGBoost*. Fitur *method_GET* menunjukkan *importance* tertinggi dengan gain score 0.4342, berkontribusi 43.42% terhadap total *information gain* model. Dominasi fitur ini mengkonfirmasi bahwa metode *HTTP* merupakan pembeda terkuat untuk membedakan permintaan normal dan anomali. Sebagian besar serangan dalam *dataset CSIC HTTP 2010* dilakukan melalui permintaan *GET* dengan injeksi *payload* pada *URL* atau *query parameters*.

Fitur *url_path_depth* (0.1091) menempati posisi kedua, menunjukkan bahwa struktur hierarkis dari URL path memberikan informasi penting. Serangan *path traversal* dan *directory enumeration* memiliki karakteristik kedalaman *path* yang berbeda dari navigasi *legitimate*. Fitur *url_has_encoding* (0.0882) di peringkat ketiga memvalidasi hasil analisis korelasi, mengkonfirmasi bahwa keberadaan *encoding* merupakan indikator kuat untuk anomali. *XGBoost* memberikan bobot tinggi pada fitur ini karena keberadaan *encoding* merupakan fitur biner yang jelas dalam memisahkan lalu lintas berbahaya dan jinak.

Fitur *sqli_total* (0.0755) sebagai skor agregat dari pola *SQL Injection* menunjukkan *importance substansial*, mengindikasikan bahwa akumulasi banyak indikator *SQL Injection* lebih informatif dibanding deteksi pola individual. Fitur struktur *URL* mendominasi 10 peringkat teratas dengan 6 dari 10 posisi teratas: *url_param_count* (0.0647), *url_numeric_ratio* (0.0515), *url_entropy* (0.0435), *url_length* (0.0234), dan *url_special_chars* (0.0104). Dominasi ini memvalidasi hipotesis bahwa karakteristik struktural dan statistik URL merupakan fondasi yang solid untuk deteksi anomali.

Fitur *pattern_idor_pattern* (0.0373) menunjukkan *importance* yang moderat meskipun *dataset* tidak memiliki sampel *IDOR* eksplisit. Hal ini mengindikasikan bahwa pola *direct object reference* dengan ID numerik muncul dalam serangan lain dan digunakan model sebagai sinyal tambahan. Fitur metode HTTP secara kolektif (*method_GET, method_POST dengan importance 0.0373, dan method_PUT dengan 0.0042)* berkontribusi 48.57% dari total *importance*, mengkonfirmasi bahwa metode permintaan merupakan fitur kategorikal yang sangat kuat untuk klasifikasi keamanan *web*.

Fitur-fitur deteksi pola seperti *pattern_sqli_quotes* (0.0046), *pattern_cmd_executables* (0.0023), *pattern_sqli_operators* (0.0009), dan *pattern_sqli_comments* (0.0001) menunjukkan importance yang relatif rendah. Namun, skor agregat seperti *total_attack_patterns* (0.0089) dan *cmd_total* (0.0038) menunjukkan bahwa kombinasi pola lebih berguna dibanding deteksi individual. Beberapa fitur seperti *idor_total* dan *redirect_total* menunjukkan *importance* 0.0000, konsisten dengan fakta bahwa *dataset* tidak memiliki sampel eksplisit untuk kategori tersebut.

Kombinasi antara fitur statistik (*entropy*, rasio *numerik*), fitur struktural (panjang, kedalaman, jumlah parameter), dan fitur berbasis pola (deteksi *SQL injection*, deteksi *encoding*) menghasilkan *ensemble* yang *robust*, dimana kelemahan dari satu kategori dikompensasi oleh kekuatan kategori lain, menciptakan pendekatan berlapis untuk deteksi anomali.

3.4 Perbandingan dengan Penelitian Terdahulu

Untuk memvalidasi kontribusi penelitian, dilakukan perbandingan performa dengan studi terdahulu pada *dataset CSIC HTTP 2010*. Tabel 4 menunjukkan hasil komparasi dengan penelitian yang memiliki metodologi *comparable*.

This is an open access article under the CC–BY-SA license

Submitted: **23/12/2025**; Accepted: **30/01/2025**; Published: **02/04/2026**

JURNAL MEDIA INFORMATIKA [JUMIN]
Volume 7 Nomor 2 April 2026, Page 245-256
ISSN 2808-005X (media online)
Available Online at http://ejournal.sisfokomtek.org/index.php/jumin

Tabel 4. Perbandingan Performa pada Dataset CSIC HTTP 2010

Peneliti (Tahun)	Algoritma	Paradigma	Acc (%)	Prec (%)	Recall (%)	F1 (%)	AUC (%)
Liang et al. (2017)	LSTM-RNN	Supervised	98.42	-	97.56	-	-
Montes et al. (2021)	RoBERTa + OCSVM	One-Class	-	-	47.10	-	
Wu et al. (2021)	AEDRAD	One-Class	-	-	-	-	95.27
Penelitian Ini	XGBoost + OWASP	Supervised	82.77	71.29	97.15	82.24	93.29

Model yang diusulkan mencapai *recall* 97.15%, setara dengan Liang et al. (97.56%) yang menggunakan LSTM-*RNN* namun dengan efisiensi komputasi lebih tinggi (5.04 detik untuk *5-fold CV*). Dibandingkan pendekatan *one-class* Montes et al. (recall 47.10%), model *supervised* ini menunjukkan peningkatan 106%, mengkonfirmasi superioritas *supervised learning* ketika labeled data tersedia. ROC-AUC 93.29% hanya ~2% di bawah *AEDRAD* (95.27%), dengan *trade-off* kompleksitas komputasi yang jauh lebih rendah.

Precision 71.29% memang lebih rendah dari beberapa studi, namun *trade-off* ini *justified* dalam konteks keamanan di mana *missed detection (false negative)* memiliki konsekuensi lebih kritis daripada *false* alarm[9][4]. *Cost-sensitive learning* (scale_pos_weight=1.436) sengaja memprioritaskan *recall* tinggi, sejalan dengan prinsip *"fail-safe"* pada sistem *WAF* modern[3]. *F1-score* 82.24% menunjukkan keseimbangan yang *reasonable* untuk *deployment* praktis[14].

Keunggulan metodologis penelitian ini terletak pada:

a. integrasi domain *knowledge* melalui 33 fitur berbasis *OWASP Top 10* yang meningkatkan interpretabilitas
b. efisiensi komputasi untuk *rapid retraining*.
c. *reproducibility* tinggi dengan *parameter* yang eksplisit. Model menawarkan solusi praktis sebagai *additional protection layer* pada *WAF* untuk meningkatkan *coverage* terhadap kategori kerentanan *OWASP Top 10*, dengan catatan bahwa validasi lebih lanjut pada *dataset* kontemporer dan lalu lintas *real-world* diperlukan untuk memastikan efektivitas terhadap varian serangan terkini[5].

3.5 Implikasi Praktis dan Limitasi

Sistem yang dikembangkan dapat diimplementasikan sebagai *additional* layer pada *Web Application Firewall (WAF)* existing untuk meningkatkan *detection coverage.* Dengan *recall rate* 97,15%, sistem mampu menangkap hampir semua serangan yang melewati *rule-based WAF* tradisional, memberikan *defense-in-depth protection*[3][4]. Model dapat di-*deploy* dalam mode monitoring untuk menganalisis *traffic patterns* dan memberikan alert kepada *security team* tanpa memblokir *request* secara otomatis. Pendekatan ini memungkinkan validasi performa dalam *production environment* sebelum mengaktifkan *blocking mode*, meminimalkan risiko *disruption* terhadap *legitimate users*.

Feature importance analysis memberikan insights tentang karakteristik serangan yang dapat digunakan untuk memperbaiki security policies dan *input validation rules*[11]. *Security team* dapat fokus pada hardening aplikasi terhadap *attack vectors* yang paling sering muncul berdasarkan *pattern detection statistics*. Sistem juga dapat berfungsi sebagai *anomaly-based IDS* yang complementary terhadap *signature-based detection*, mampu mengidentifikasi *zero-day attacks*[4], atau attack variations yang belum memiliki signature.

Meskipun menunjukkan hasil yang menjanjikan, penelitian ini memiliki beberapa keterbatasan yang perlu diakui. *Dataset CSIC HTTP 2010* tidak mengandung sampel eksplisit untuk kategori *Open Redirect* dan *IDOR*, sehingga deteksi untuk kedua kategori tersebut bergantung pada pola anomali umum yang mungkin kurang spesifik. Tingkat *false positive* sebesar 27.24% dapat menjadi perhatian dalam *deployment* produksi, karena pemblokiran lalu lintas *legitimate* berpotensi berdampak pada pengalaman pengguna dan operasi bisnis. Namun demikian, *trade-off* ini masih dapat diterima mengingat tingkat *recall* yang sangat tinggi (97.15%) serta nilai *ROC-AUC* yang sangat baik (93.29%)[9], yang menunjukkan bahwa model mampu mengoptimalkan cakupan keamanan dengan tingkat alarm palsu yang relatif terkendali. Model dilatih dan diuji menggunakan dataset yang dikumpulkan pada tahun 2010, sehingga mungkin tidak sepenuhnya merefleksikan lanskap serangan modern yang terus berkembang[5]. Oleh karena itu, pengujian pada dataset yang lebih baru seperti Bot-IoT[15] atau lalu lintas *real-world* diperlukan untuk memvalidasi kemampuan generalisasi model terhadap variasi serangan dan teknik eksploitasi baru[5][16]. Evolusi teknologi *web* dan metodologi serangan dalam lebih dari satu dekade terakhir juga menuntut pelatihan ulang model secara berkelanjutan dengan data kontemporer guna mempertahankan efektivitas deteksi.

4. KESIMPULAN

Penelitian ini berhasil mencapai tujuan pengembangan sistem deteksi kerentanan keamanan aplikasi *web* berbasis *machine learning* dengan mengintegrasikan algoritma *XGBoost*[6] dan pendekatan *pattern recognition* berbasis *OWASP*

This is an open access article under the CC–BY-SA license

Terakreditasi SINTA 5 SK :72/E/KPT/2024
Submitted: **23/12/2025**; Accepted: **30/01/2025**; Published: **02/04/2026**

Top 10[2]. Pendekatan yang diusulkan mampu mengidentifikasi serangan web secara efektif, khususnya pada kategori kerentanan yang memiliki representasi eksplisit dalam *dataset* pelatihan, sehingga menunjukkan potensi sebagai mekanisme deteksi yang andal dan dapat berfungsi sebagai lapisan tambahan pada *Web Application Firewall* dalam skema pertahanan berlapis[3]. Meskipun demikian, penelitian ini memiliki keterbatasan pada tingginya tingkat *false positive* serta keterbatasan cakupan untuk beberapa kategori kerentanan yang tidak tersedia secara eksplisit dalam *dataset*, sehingga hasil deteksi pada kategori tersebut masih bergantung pada pola anomali umum. Oleh karena itu, penelitian selanjutnya disarankan untuk memfokuskan pada pengurangan *false positive* melalui mekanisme validasi tambahan atau penyesuaian ambang keputusan, penggunaan *dataset* yang lebih mutakhir atau lalu lintas nyata untuk meningkatkan relevansi terhadap serangan web modern, serta optimisasi sistem agar dapat diterapkan secara *real-time* pada lingkungan dengan lalu lintas tinggi[14].

UCAPAN TERIMAKASIH

Terima kasih disampaikan kepada pihak-pihak yang telah mendukung terlaksananya penelitian ini.

REFERENCES

[1] S. Saeed, A. Alzahrani, and R. Khan, “Digital transformation and cybersecurity challenges for businesses resilience,” *Sensors*, vol. 23, no. 15, p. 6666, 2023, doi: 10.3390/s23156666.

[2] OWASP Foundation, “OWASP Top 10 - 2021: The ten most critical web application security risks,” 2021. [Online]. Available: https://owasp.org/Top10/

[3] B. R. Dawadi, B. Adhikari, and D. K. Srivastava, “Deep learning technique-enabled web application firewall for the detection of web attacks,” *Sensors*, vol. 23, no. 4, p. 2073, 2023, doi: 10.3390/s23042073.

[4] A. Khraisat, I. Gondal, P. Vamplew, and J. Kamruzzaman, “Survey of intrusion detection systems: techniques, datasets and challenges,” *Cybersecurity*, vol. 2, no. 1, p. 20, 2019, doi: 10.1186/s42400-019-0038-7.

[5] D. Chou and M. Jiang, “A survey on data-driven network intrusion detection,” *ACM Comput. Surv.*, vol. 54, no. 9, pp. 1–36, 2021, doi: 10.1145/3472753.

[6] T. Chen and C. Guestrin, “XGBoost: A scalable tree boosting system,” *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 785–794, 2016, doi: 10.1145/2939672.2939785.

[7] F. Ullah, S. Ullah, G. Srivastava, and J. C. W. Lin, “Enhancing structured query language injection detection with trustworthy ensemble learning and boosting models using local explanation techniques,” *Electronics (Basel).*, vol. 13, no. 22, p. 4350, 2024, doi: 10.3390/electronics13224350.

[8] C. Torrano-Gimenez, A. Perez-Villegas, and G. Alvarez, “An anomaly-based approach for intrusion detection in web traffic,” *Journal of Information Assurance and Security*, vol. 5, no. 4, pp. 446–454, 2010, [Online]. Available: https://www.tic.itefi.csic.es/dataset/

[9] Z. Ahmad, A. Shahid Khan, C. Wai Shiang, J. Abdullah, and F. Ahmad, “Network intrusion detection system: A systematic study of machine learning and deep learning approaches,” *Transactions on Emerging Telecommunications Technologies*, vol. 32, no. 1, p. e4150, 2021, doi: 10.1002/ett.4150.

[10] K. Tadhani, D. Shah, D. Garg, M. Slamy, M. Shah, and V. Vijayakumar, “Securing web applications against XSS and SQLi attacks using a novel deep learning approach,” *J. Electron. Imaging*, vol. 33, no. 1, p. 11805, 2024, doi: 10.1117/1.JEI.33.1.011805.

[11] V. Hnamte and J. Hussain, “A comparative study of using boosting-based machine learning algorithms for IoT network intrusion detection,” *International Journal of Computational Intelligence Systems*, vol. 16, no. 1, p. 194, 2023, doi: 10.1007/s44196-023-00355-x.

[12] H. Liu, Y. Zhang, and J. Wang, “Deep learning in cybersecurity: A hybrid BERT-LSTM network for SQL injection attack detection,” *IET Inf. Secur.*, vol. 18, no. 5, p. 565950, 2024, doi: 10.1049/2024/5565950.

[13] Z. Thalji, A. Al-Saedi, S. Al-Amidi, and N. Al-Jamali, “AE-Net: Novel autoencoder-based deep features for SQL injection attack detection,” *Computers*, vol. 12, no. 5, p. 100, 2023, doi: 10.3390/computers12050100.

[14] F. Gouveia and M. Correia, “Network intrusion detection with XGBoost and deep learning algorithms: An evaluation study,” in *IEEE International Conference on Cyber Security and Resilience (CSR)*, 2021. doi: 10.1109/CSR54599.2021.9457953.

[15] N. Koroniotis, N. Moustafa, E. Sitnikova, and B. Turnbull, “Towards the development of realistic botnet dataset in the Internet of Things for network forensic analytics: Bot-IoT dataset,” *Future Generation Computer Systems*, vol. 100, pp. 779–796, 2020, doi: 10.1016/j.future.2019.05.041.

[16] T. T. Nguyen and V. J. Reddi, “Deep reinforcement learning for cyber security,” *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 34, no. 8, pp. 3779–3795, 2023, doi: 10.1109/TNNLS.2021.3121870.

This is an open access article under the CC–BY-SA license