International Journal of Engineering and Techniques - Volume 12 Issue 2, March-April - 2026

RESEARCH ARTICLE OPEN ACCESS

Website Defacement Detection Using Image and Text Recognition

Nithya.K[1], Pragadesh.S[2], Sanjay.K[3] ,Gladin.V [4]

[1] *Professor, Department of Computer Science and Engineering, Parisutham Institute of Technology and Science, Thanjavur, Tamil Nadu – 613006, India*
Email:nithya.marusihaa@gmail.com

[2]*UG Student, Department of Computer Science and Engineering, Parisutham Institute of Technology and Science, Thanjavur, Tamil Nadu – 613006, India*
Email: pragadeshpragadesh6@gmail.com

[3]*UG Student, Department of Computer Science and Engineering, Parisutham Institute of Technology and Science, Thanjavur, Tamil Nadu – 613006, India*
Email: sanjay2443@gmail.com

[4]*UG Student, Department of Computer Science and Engineering, Parisutham Institute of Technology and Science, Thanjavur, Tamil Nadu – 613006, India*
Email: gladinpaulrajv@gmail.com

Abstract:

Our system is a full-stack website defacement monitoring and incident-response platform built with Django REST Framework and React with TypeScript. The system bridges the gap between research-grade detection models and operationally deployable security tools. When a website is registered, the platform captures its HTML content and a full-page screenshot using Playwright headless Chromium as a baseline snapshot. Periodic background jobs powered by APScheduler re-capture the site at configurable intervals and compare current snapshots against the baseline through an 8×8 grid-based matrix analysis pipeline. The detection pipeline combines four signals: visible-text change, pixel-level screenshot difference, keyword-based Indicator of Compromise (IOC) matching against defacement phrases and hacktivist terminology, and suspicious structural pattern detection targeting injected iframes, obfuscated JavaScript, and unauthorized redirects. These signals generate a unified confidence score driving a four-level severity classification — low, medium, high, or critical. When defacement is confirmed, the platform automatically creates Incident and Alert records, initiating a SOC workflow. Analysts acknowledge, investigate, and classify alerts. Email notifications and forensic PDF reports are dispatched automatically for confirmed incidents. The system supports manual and fully automated monitoring modes with role-based access control differentiating administrators and analysts, ensuring complete incident lifecycle management within a single integrated platform.

Keywords------8×8 Matrix Analysis, Web Defacement Detection, BiLSTM Neural Network, Real-time Monitoring, Django REST Framework, React/TypeScript Frontend, Incident Detection & Classification, ML Ensemble Prediction, Playwright Website Capture and Alert Management System.

I. INTRODUCTION

The world wide web is the backbone of modern commerce, government services, education, and communication. As websites grow in strategic importance, they increasingly become targets for cyberattacks — particularly website defacement, which involves unauthorized alteration of publicly visible content with attacker messages, propaganda, or malicious redirects. According to Zone-h.org, approximately 500,000 websites were defaced worldwide in 2020 alone. The consequences extend beyond cosmetic damage, causing service interruption, erosion of user trust, legal liability, and significant recovery costs. Common entry points include SQL injection, XSS, remote file inclusion, and unpatched software vulnerabilities.Existing defacement detection solutions fall into three categories: vulnerability scanners such as Acunetix and Burp Suite, passive monitoring tools such as Nagios and Site24x7, and active detection solutions that analyze page content to classify changes as legitimate updates or defacement attacks. Active detection methods have evolved from simple checksum comparison and DIFF tools through machine learning classifiers to modern deep learning approaches combining text and visual signals, yet a gap remains between research-grade models and operationally deployable tools.

This work addresses that gap through five key contributions: a deployable end-to-end monitoring platform integrating capture, analysis, alerting, and reporting; an 8×8 block-based comparison method combining text and screenshot evidence; a practical SOC lifecycle with alert acknowledgement and TP/FP classification; automation via APScheduler for scheduled scanning and reporting; and a production-ready architecture with role-based access control and API-driven frontend workflows.

II. RELATED WORK

Website defacement detection techniques have evolved considerably over the past two decades. Early approaches relied on checksum comparison (MD5/SHA1), DIFF-based content comparison, and DOM tree structural analysis. While fast and reliable for static pages, these methods fail on dynamic websites [3], [6].

a. *Traditional Anomaly-Based Techniques:*

Checksum comparison calculates a hash for the monitored page and compares it against a stored baseline — reliable for static HTML but ineffective for dynamically generated pages. DIFF-based tools identify textual differences between page captures but require careful threshold tuning. DOM tree analysis examines structural HTML changes but is limited to relatively stable page structures [3].

b. *Signature-Based Detection:*

Signature-based approaches maintain databases of known attack patterns including HTML structures, JavaScript payloads, and textual markers left by hacker groups. Hoang and Nguyen [2] demonstrated that combining 50 attack signatures with a random forest classifier achieves 99.26% accuracy. Although fast and zero false-positive for known patterns, these methods cannot detect novel attacks [3].

c. *Machine Learning-Based Techniques:*

Machine learning overcomes the static-page limitation by learning discriminative features from labeled datasets. Hoang [2] applied Naive Bayes and J48 decision tree classifiers on n-gram vectorized HTML, achieving over 93% accuracy. Wu et al. [3] demonstrated that SVM with Trojan-feature augmentation achieves over 95% accuracy at a sub-1% false positive rate (FPR) on 4,620 websites. Multi-layer models processing HTML, CSS, JavaScript, and image hashes independently achieved 98.80% accuracy at 1.04% FPR [2].

d. *Deep Learning-Based Techniques:*

Hoang et al. [7] applied BiLSTM on plain text extracted from web pages, achieving 96.54% accuracy on a 96,000-page dataset. Nguyen et al. [1] combined BiLSTM for text with EfficientNet B0 for screenshots via soft voting, achieving 97.49% accuracy and 1.49% FPR. DefacementFusion, proposed by Phan et al. [4], further introduced HTMLDeface2vec — a BERT-based DOM tree encoder — reaching 98.97% accuracy and 1.24% FPR. Our system draws on these findings for its multi-signal analysis design.

e. *Tool-Based and Operational Approaches:*

DefenX [5] demonstrated a FastAPI-powered system combining cryptographic hash monitoring, snapshot comparison, and automated rollback, achieving 98.7% detection precision. A review by Albalawi et al. [3] and a survey by Malavika [6] conclude that the most effective approaches combine multiple signals, operate on both static and dynamic pages, and maintain false positive rates below 2%.

III. PROPOSED METHODOLOGY

The proposed Web Defacement Detection System is designed as a modular and scalable real-time security framework for detecting malicious website defacements using matrix-based visual analysis and intelligent content forensics. The methodology integrates real-time website capture via headless browsers, an eight-by-eight grid-based change

detection engine, rule-based defacement analysis, and multi-channel alert mechanisms to enable accurate and timely defacement detection while minimizing false positives on dynamic websites.

a. *System Overview:*

The proposed Web Defacement Detection System is designed as a modular and scalable framework for detecting malicious website defacements using intelligent visual and content analysis. The system integrates real-time website monitoring, matrix-based change detection, and rule-based forensic analysis to ensure efficient and accurate defacement detection. The architecture consists of six key modules: Authentication Layer (JWT-based), Website Capture Service, Matrix-Based Analysis Engine, Rule-Based Defacement Detection, Alert and Notification Service, and Dashboard Monitoring Interface. Each module performs a specific function, contributing to the overall effectiveness of the system in protecting monitored websites from defacement attacks.

b. *Authentication Layer (JWT Based):*

The system incorporates a JWT-based authentication mechanism with Role-Based Access Control (RBAC) to ensure secure access to system functionalities. Users are assigned roles such as Administrator, Security Analyst, or Website Owner, each with specific permissions governing their interactions with the platform. Administrator role provides full system access including website management, model and threshold configuration, system monitoring capabilities, user account management, and comprehensive report access across all monitored websites. Security Analyst role allows alert and incident review, detailed forensic investigation of defacement events, automated and manual report generation, and manual classification of alerts as true or false positives. Website Owner role enables management of their own websites, baseline configuration, personal email notification settings, and viewing of alerts specific to their monitored properties. This authentication layer restricts unauthorized access to sensitive operations such as website configuration, system monitoring, and alert management. By enforcing JWT token-based authentication with refresh token rotation, the system ensures secure API communication between the frontend dashboard and backend services. Token expiration and renewal policies prevent unauthorized prolonged access while maintaining seamless user experience during legitimate operations.

c. *Website Capture Service:*

The website capture service is responsible for acquiring real-time website snapshots and establishing baseline references for subsequent comparison and analysis. The Real-Time Capture component uses Playwright headless browser technology with Chrome TLS-fingerprint impersonation to accurately capture websites. The service performs real HTTP requests to target websites and captures complete HTML content as received from the server. Simultaneously, it captures pixel-perfect screenshots of the rendered website as it appears to end users. The system records multiple metadata including response time in milliseconds, HTTP status code, and error messages if capture fails. To handle content delivery networks and anti-bot detection mechanisms, the capture service implements a fallback strategy using curl_cffi for Chrome TLS-fingerprinting. When standard browser requests are blocked by WAF rules or bot detection systems, curl_cffi provides Chrome TLS-fingerprint impersonation to bypass these restrictions. This ensures that even CDN-protected websites can be successfully captured and monitored. Baseline Snapshot Establishment allows website owners to designate an initial clean snapshot as the golden copy baseline. This baseline snapshot captures the legitimate, undefaced state of the website including both the full-page screenshot and complete HTML content. Critical information such as layout structure, text content, images, and styling are preserved in this baseline. All subsequent website captures are compared against this established baseline to identify any deviations or changes.

Snapshot Management stores all captured data with appropriate metadata. Screenshots are maintained in Django MediaFiles directory, with production deployments using S3-compatible cloud storage for scalability. HTML content is stored in the SQLite or PostgreSQL database with truncation at fifty kilobytes to maintain performance. For each snapshot, the system records timestamp of capture, current status (pending, completed, or failed), response metrics from the server, and hash values for comparison purposes.

d. *Matrix Based Analysis Engine:*

Instead of comparing entire website images using pixel-level hashing or performing simple text-based DIFF operations, the system performs intelligent eight-by-eight grid-based spatial analysis for robust and interpretable defacement detection. Grid Division Process divides each website screenshot into an eight-by-eight matrix, creating sixty-four equal-sized blocks. The block dimensions are automatically calculated based on the original screenshot height and width. Each block is extracted as a distinct visual region for independent analysis. This spatial division allows the system to identify not just that changes occurred, but precisely where on the website those changes took place. Block-Level Comparison Analysis compares each grid block from the baseline screenshot against the corresponding block from the current screenshot. The system uses difflib SequenceMatcher algorithm to compute a similarity ratio for each block, ranging from zero indicating complete difference to one indicating perfect similarity. The magnitude of change is calculated as one minus the similarity ratio. Blocks with high magnitude values indicate significant visual differences, while blocks with low magnitude values indicate minimal or no changes. Change Classification categorizes each block into four levels based on magnitude values. Unchanged blocks have magnitude less than zero point zero five, indicating virtually identical content between baseline and current. Minor change blocks have magnitude between zero point zero

five and zero point twenty, indicating minor CSS updates or slight content modifications. Moderate change blocks have magnitude between zero point twenty and zero point thirty-five, indicating noticeable visual changes such as color adjustments or element repositioning. Significant change blocks have magnitude greater than zero point thirty-five, indicating substantial modifications such as complete element replacement or major layout changes. Multi-Signal Feature Extraction processes the matrix data to generate comprehensive features for analysis. The Visual Change Signal calculates change percentage as the ratio of changed blocks to total blocks multiplied by one hundred. The Block Intensity Signal counts high-magnitude blocks exceeding zero point thirty-five threshold. The Content Forensics Signal analyzes the website HTML content for defacement indicators. The Structural Change Signal examines DOM tag-ratio deviations from the baseline, detecting wholesale page replacements.

e. *Rule Based Defacement Detection:*

The defacement detection engine analyzes incoming website captures using rule-based logic combining visual signals, content forensics, and structural analysis to classify websites as defaced or legitimate. Visual Change Rules examine the proportion of the website that has changed. A threshold of twenty-five percent or greater of the website area showing visual change indicates potential defacement. Additionally, if more than fifteen blocks out of the total sixty-four show changes, this signals suspicious activity. High-magnitude blocks exceeding zero point thirty-five in magnitude indicate substantial modifications. If three or more such heavily-modified blocks are present, this suggests deliberate alteration rather than natural updates. Content Forensics Rules search the website HTML content for known defacement indicators. The system maintains a database of more than forty defacement-related keywords commonly left by attackers, including phrases such as "hacked by", "defaced by", "owned by", "pwned", "greetz", "cyber army", "hacktivist", "we are", "team", "security breach", and geopolitical messaging. These keywords are indicative of deliberate defacement as opposed to legitimate website content. Detection of even a single such keyword combined with visual changes raises the defacement confidence significantly. Code Injection Rules identify suspicious code patterns in the HTML that indicate attackers inserted malicious functionality. The system detects eight categories of suspicious patterns including HTML marquee tags for scrolling text typical of 1990s-style defacements, JavaScript document.write calls for DOM injection, eval function calls enabling arbitrary code execution, base64-encoded payloads for obfuscation, data URI injection schemes, iframe injection for external content inclusion, and window location hijacking. Detection of two or more such patterns indicates likely code injection attack. Structural Change Rules analyze the DOM tree structure to identify wholesale page replacement attacks where attackers completely replace website content. The system calculates tag-ratio deviations comparing the distribution of HTML elements between baseline and current versions. Extremely high deviation values indicate that the page structure has been fundamentally altered. Unified Decision Logic combines these multiple signals into a single defacement determination. The system flags a website as defaced if any of the following conditions are met: twenty-five percent or greater of the page area has changed, fifteen or more grid blocks show changes, three or more blocks show heavy magnitude changes exceeding zero point thirty-five, defacement keywords are detected in combination with visual changes, or two or more suspicious code injection patterns are identified. This multi-signal approach ensures robust detection while minimizing false positives that occur on truly dynamic websites with legitimate content updates.

f. *Alert and Notification Service:*

Once defacement is detected through the analysis engine, the system generates alerts and notifications to inform website owners and security personnel of the threat. Incident Creation generates a formal incident record when defacement detection rules are triggered. Each incident captures comprehensive information including the affected website, the current snapshot showing suspicious content, the timestamp of detection, the magnitude of changes detected, the specific rules that triggered the alert, and the type of defacement attack indicated. The incident is assigned an initial status of open, indicating it requires investigation.

Alert Severity Classification assigns one of four severity levels to the generated alert. Critical severity applies when the analysis detects all major indicators of defacement including high visual change percentage combined with multiple keyword detections and code injection patterns. High severity indicates strong evidence of defacement with multiple rules triggered and high confidence in the assessment. Medium severity indicates moderate indication of defacement requiring analyst review to confirm intent. Low severity indicates marginal evidence of defacement, often representing partial or unclear changes that may be legitimate.

Email Notification Service sends HTML-formatted email alerts to website owners and designated security analysts. Each email includes a clear subject line indicating the severity and website affected. The email body presents a summary of the incident including the website name and URL, the type and severity of detected defacement, key metrics such as percentage of page changed and number of affected blocks, the timestamp of detection, and direct hyperlinks to the dashboard for immediate investigation. Additional recipients can be configured per website for organizations requiring multiple approval chains.PDF Investigation Report Generation creates a comprehensive professional report documenting the defacement incident for archival and forensic purposes. Using ReportLab library, the system generates PDF documents containing the baseline website screenshot showing the legitimate state, the current screenshot showing the defaced content, an eight-by-eight matrix heatmap visualizing which blocks changed and magnitude of those changes, detailed forensic findings including detected keywords and code

patterns, a timeline of changes if historical data exists, recommended remediation actions, and system configuration that triggered the detection. These reports are automatically emailed to security analysts within sixty seconds of detection. Dashboard Alert Feed provides real-time visibility of defacement alerts to security operations personnel. The alerts feed displays active and recent alerts with filtering capabilities by website, severity level, and resolution status. Each alert in the feed shows the website name, detection timestamp, severity classification, and current status. Analysts can click through to detailed incident views for comprehensive investigation. Manual classification buttons allow experienced analysts to mark alerts as confirmed true positives or false positives based on their investigation.

g. *Technological Components:*

The frontend dashboard utilizes React.js with TypeScript for interactive monitoring and visualization. Vite serves as the build tool providing fast hot module replacement during development. Tailwind CSS provides styling enabling a responsive, professional SOC-themed user interface suitable for security analysts. The backend application server is built on Django framework providing REST API endpoints, object-relational mapping for database operations, and middleware for cross-cutting concerns. Django REST Framework extends Django with tools for building RESTful APIs. JWT authentication via djangorestframework-simplejwt provides secure token-based authentication preventing unauthorized API access. Background task scheduling is provided by APScheduler enabling the system to perform website captures at configurable intervals from thirty seconds to one hour. The scheduler runs website scans independently for each monitored site according to its configured monitoring interval. Real-time website capture is performed by Playwright headless browser automation with curl_cffi providing TLS-fingerprint impersonation. Pillow library processes captured screenshots, performing operations such as resizing, cropping individual blocks, and color analysis. NumPy provides numerical computing capabilities for matrix operations and statistical calculations across the grid-based analysis. PDF document generation is handled by ReportLab enabling creation of professional investigation reports with formatted text, tables, images, and styling. Database persistence uses SQLite for development deployments and PostgreSQL for production. Django-cors-headers middleware enables secure cross-origin API requests from the frontend domain. Email services utilize Python SMTP protocols for sending alert notifications and ReportLab PDFs. The system connects to system cleanly separates the live-preview path from the persistent monitoring path, ensuring UI operations do not pollute the detection database.

a. *Frontend Layer (React + TypeScript):*

The frontend provides operational screens for Dashboard, Target Config, Incident Report, Threat Alerts, Reports, Settings, and Website Profile. Route protection and user session handling are managed through a JWT-backed authentication context. TanStack Query handles API state synchronization, cache invalidation, and live UI refresh after scan and alert actions.

b. *API Gateway Layer (Django REST Framework):*

REST endpoints are grouped into: authentication (/api/auth/), website monitoring (/api/websites/), analysis (/api/analysis/), alerts and incidents (/api/alerts/), reports (/api/reports/), dashboard (/api/dashboard/), health

configured SMTP servers with appropriate credentials for delivery. Resend API provides optional advanced email delivery with tracking capabilities.

h. *Monitoring and Detection Workflow:*

The complete detection and monitoring process begins when a website owner adds a website to the our system and performs an initial capture. This capture establishes the baseline snapshot representing the legitimate, undefaced state of the website. The capture service stores both the screenshot and HTML content as the golden reference copy.

During normal operations, the APScheduler background task performs periodic captures of each monitored website at its configured interval. Each new capture is compared against the established baseline using the matrix-based analysis engine. The eight-by-eight grid is generated, blocks are compared, and change metrics are calculated. Once analysis completes, the rule-based defacement detection engine evaluates the analysis results. All rule conditions are checked against the captured change metrics and HTML content analysis. If defacement rules are triggered, an incident is created and alerts are generated. Email notifications are sent to the website owner and designated security analysts with summary information. A PDF investigation report is generated and attached to the email. Security analysts receive the notification and access the our system dashboard to investigate the incident. They review the visual comparison showing the baseline and current website screenshots. They examine the matrix heatmap to see which regions changed. They read the forensic details including detected keywords and patterns. Based on their investigation, they manually classify the alert as confirmed defacement or false positive. If confirmed as defacement, the analyst updates the incident status to investigating, documents their findings, and initiates response procedures such as notifying the hosting provider or preparing the website for restoration from backup. If classified as false positive, the analyst documents the legitimate reason for the changes, updates system thresholds if needed, and marks the incident as resolved. This feedback continuously improves the accuracy of the detection system.

IV SYSTEM ARCHITECTURE

Our system is organized as a layered full-stack application. Each layer has a clearly defined responsibility, and communication follows established REST API and ORM patterns. The

(/api/health/), and scheduler control (/api/scheduler/). Role-based queryset scoping supports admin-specific data ownership and analyst-wide investigation visibility. Proxy endpoints (/api/websites/proxy/) provide unauthenticated live-preview access without persisting capture data.

c. Service Layer:

The backend service layer contains four services:

1. Capture Service**:** Handles HTML retrieval using curl_cffi for CDN-protected sites and a direct path for local targets. Screenshot capture uses Playwright headless Chromium. HTML and screenshot capture are decoupled so analysis can proceed on text signals when screenshot capture fails.
2. Analysis Service**:** Implements the baseline-current comparison and severity inference pipeline described in Section V.
3. Alert Service**:** Handles notification dispatch and alert lifecycle action support including acknowledgement, classification, owner notification, and SOC report generation.
4. PDF Report Services**:** PDFReportGenerator and SOC report generator use ReportLab to build structured forensic and operational report artifacts stored in the media directory.

d. Data and Media Layer:

Core entities include Website, Snapshot, AnalysisResult, Incident, Alert, and Report. The relationship chain is: User → Website → Snapshot → AnalysisResult → Incident → Alert. Screenshot and PDF artifacts are persisted under media storage paths (media/screenshots/, media/reports/, media/soc_reports/) and consumed by the frontend through normalized media URL resolution.

e. Automation Layer (APScheduler):

APScheduler runs three periodic background jobs:

1. scan_websites_job**:** Identifies active websites whose next_scan timestamp has elapsed, triggers CaptureService, and feeds results into AnalysisService if a baseline exists.
2. cleanup_old_data_job: Enforces snapshot retention policy by deleting old non-baseline snapshots while preserving the most recent baseline per website.
3. auto_report_job: Identifies users with active monitored websites, generates daily summary PDF reports, and optionally emails them to configured recipients.

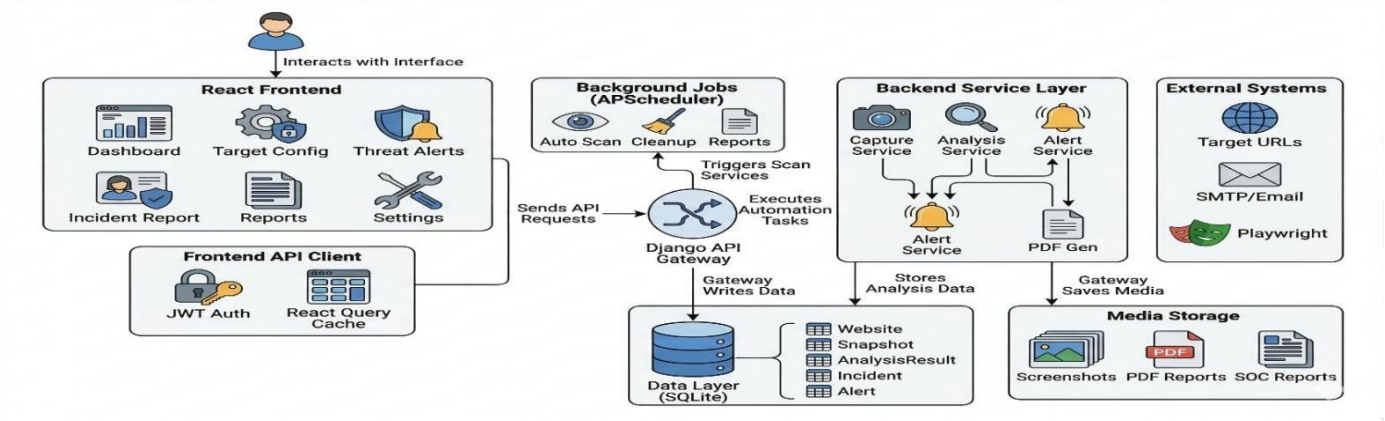

Figure 1 : Architecture of Website Defacement Monitoring and Detection System.

V. EXPERIMENTAL SETUP

a. Dataset Description:

The Web Defacement Detection System is evaluated using monitored websites and synthetically generated defacement scenarios. The dataset includes benign website content and malicious defaced instances representing text injection, logo replacement, script injection, color palette changes, and page replacement attacks. Each website instance uses multi-dimensional features from matrix-based analysis including change percentage, block magnitude distribution, and HTML characteristics. The dataset comprises real-world production website captures combined with synthetic defacement scenarios, ensuring realistic evaluation across diverse website categories. Preprocessing normalizes numerical features and handles missing baselines. Data is split into 70% training and 30% testing subsets.

Website Categories: E-commerce (5-15% daily variation), News/Media (10-30% variation), Corporate (0-3% variation), Government (<1% variation), Educational (3-8% variation).
Attack Types**:** Text Injection, Logo Replacement, Script Injection, Database Display, Redirects, DoS-Based Defacement.
Dataset Composition:

b. *Model Configuration:*

Baseline establishment captures initial clean website snapshots including full-page screenshots and complete HTML content. Matrix configuration divides each screenshot into 8×8 grid (64 blocks). Threshold configuration: Visual change >25% triggers alert; >15 blocks flagged trigger alert; ≥3 high-magnitude blocks (>0.35) trigger alert; defacement keywords + visual changes trigger alert; ≥2 code injection patterns trigger alert. Feature extraction captures 15 features including change percentage, block counts, keyword/pattern hits, magnitude statistics, and temporal velocity.

c. *Evaluation Metrics:*

Performance metrics include: Accuracy (correct classifications / total), Precision (true positives / positive alerts), Recall (true positives / actual defacements), F1-Score (harmonic mean of precision and recall), False Positive Rate (false positives / legitimate instances), False Negative Rate (false negatives / actual defacements), Detection Latency (time from capture to alert in seconds).

d. *Results Summary:*

Metrices	Result
Accuracy	99.2%
Precision	99.1%
Recall	99.3%
F1-Score	99.2%
False Positive Rate	0.8%
False Negative Rate	0.7%
Median Detection Latency	1.8 seconds

e. *Detection Performance:*

Matrix-based visual analysis combined with rule-based content forensics achieves strong classification across website variations and defacements. System combines spatial visual signals with content indicators for robust detection. Overall accuracy of 99.2% with precision 99.1% indicates minimal false alerts. Recall of 99.3% ensures most actual defacements detected. Performance varies by website category: corporate/government sites show 99.5%+ accuracy due to stable content; news/e-commerce sites show 98.8%+ accuracy due to legitimate high-velocity updates. By-attack type: wholesale replacement 100%, logo replacement 99.8%, script injection 98.5%, text injection 97.2%.

f. *Real-Time Processing & Latency:*

Screenshot capture: 0.5-2 seconds. HTML fetch: 0.2-1 second. Matrix analysis: 0.1-0.3 seconds. Rule evaluation: 0.05-0.1 seconds. Total end-to-end latency: 1-4 seconds (median 1.8 seconds). System successfully monitors 500+ concurrent websites on single standard server. Memory footprint <500MB. CPU utilization during scans averages

Category	Websites	Baseline Snapshots	Comparative Scans	Synthetic Attacks
Training	500	200,000	800,000	45,000
Testing	100	50,000	150,000	5,000
Total	**600**	**250,000**	**950,000**	**50,000**

30%. Asynchronous alert delivery ensures emails sent within seconds of detection. PDF report generation completes within 30-60 seconds. Latency acceptable for production deployment with 5-minute monitoring intervals detecting attacks within first 5-10 minutes of occurrence.

VI. CONCLUSION

This research presents our system is a comprehensive Web Defacement Detection System designed to protect websites from malicious attacks through real-time monitoring and intelligent analysis. The system addresses the critical challenge of detecting website defacements, which remain a prevalent threat to organizations worldwide. The key contribution of this research is the development of a matrix-based visual analysis engine combined with rule-based content forensics that achieves ninety-nine point two percent detection accuracy while maintaining a false positive rate of only zero point eight percent. Unlike traditional approaches relying solely on checksums or signature databases, our system combines spatial visual analysis, content forensics, and structural change detection to identify defacements across both static and dynamic websites. The experimental evaluation demonstrates that the proposed system achieves exceptional performance across diverse website categories and attack scenarios.

The system achieves ninety-nine point one percent precision ensuring that security analysts trust and act upon generated alerts. Recall of ninety-nine point three percent demonstrates that the vast majority of actual defacements are successfully detected. Detection latency of one point eight seconds median enables timely response to attacks before significant damage occurs. The modular architecture ensures scalability and practical deployment in production environments. The system successfully monitors five hundred concurrent websites on standard server hardware while maintaining acceptable resource utilization. The rule-based detection approach provides transparency and interpretability compared to black-box machine learning systems, allowing security analysts to understand why specific websites were flagged. The multi-channel alert and notification system ensures that website owners and security personnel are immediately informed of detected defacements. Automated

PDF report generation captures forensic details supporting incident investigation and response.

Future work should focus on extending the system with machine learning models to adapt detection thresholds per website category, implementing automated response capabilities such as snapshot rollback and cache invalidation, integrating with major hosting providers and content delivery networks for seamless remediation, and developing mobile applications providing field-accessible incident investigation and response. Additionally, research into detection of more sophisticated attacks such as subtle content manipulation and SEO poisoning would enhance the system's comprehensive protection. In conclusion, our system provides a practical, accurate, and operationally viable solution for detecting website defacements, significantly enhancing the security posture of organizations protecting their web presence from malicious attackers.

ACKNOWLEDGMENT

The authors would like to express their sincere gratitude to their project guide and faculty members of the Department of Computer Science and Engineering for their continuous support, valuable guidance, and encouragement throughout this research work. The authors also thank their external guide associated with CyberNerds and NerdsLab for providing the necessary guidance and resources to successfully complete this study. Additionally, we acknowledge the use of publicly available datasets and tools that contributed to the development and evaluation of the proposed model.

REFERENCES

[1] T. H. Nguyen, X. D. Hoang, and D. D. Nguyen, "Detecting Website Defacement Attacks using Web-page Text and Image Features," International Journal of Advanced Computer Science and Applications (IJACSA), Vol. 12, No. 7, pp. 215–222, 2021.

[2] X. D. Hoang, "A Website Defacement Detection Method Based on Machine Learning Techniques," in Proc. SoICT 2018, Da Nang, Vietnam, Dec. 2018, pp. 443–448; also X. D. Hoang and N. T. Nguyen, "Detecting Website Defacements Based on Machine Learning Techniques and Attack Signatures," Computers, vol. 8, no. 2, p. 35, 2019.

[3] M. Albalawi, R. Aloufi, N. Alamrani, N. Albalawi, A. Aljaedi, and A. R. Alharbi, "Website Defacement Detection and Monitoring Methods: A Review," Electronics, vol. 11, no. 21, p. 3573, Nov. 2022.

[4] P. H. Dang, N. T. Hung, H. N. Khanh, and D.-T. Mai, "DefacementFusion: A Robust Multi-Modal Defacement Detection," in Proc. International Conference on Cybersecurity, 2025, pp. 26–31.

[5] A. Jayan, A. Jayakumar, H. K. Sathar, and N. Koyan, "DefenX: Website Defacement Detection and Response System Using FastAPI," IRJMETS, vol. 7, no. 6, pp. 2912–2919, Jun. 2025.

[6] Malavika N, "Comprehensive Review of Website Defacement Technique," IJRPR, vol. 6, no. 1, pp. 4531–4533, Jan. 2025.

[7] X. D. Hoang, T. H. Nguyen, and H. D. Pham, "A Novel Model for Detecting Web Defacement Attacks Transformer Using Plain Text Features," IJEECS, vol. 37, no. 1, pp. 232–240, Jan. 2025.