computers

MDPI

Article

Automated OSINT Techniques for Digital Asset Discovery and Cyber Risk Assessment

Tetiana Babenko [1], Kateryna Kolesnikova [2], Olga Abramkina [3] and Yelizaveta Vitulyova [4,5,6,*]

[1] Department of Cybersecurity, International IT University, Manas Str., 34/1, Almaty 050000, Kazakhstan; babenko.tetiana.v@gmail.com
[2] Department of Information Systems, International IT University, Manas Str., 34/1, Almaty 050000, Kazakhstan; kkolesnikova@iitu.edu.kz
[3] Department of Cybersecurity, Almaty University of Power Engineering and Telecommunications Named After Gumarbek Daukeev, Baitursynuly Str., 126, Almaty 050013, Kazakhstan; olga.manank@gmail.com
[4] National Scientific Laboratory for the Collective Use of Information and Space Technologies (NSLC IST), 22 Satbayev Street, Almaty 050013, Kazakhstan
[5] JSC "Institute of Digital Engineering and Technology", 22/5 Satbayev Street, Almaty 050013, Kazakhstan
[6] Department Smart Technologies in Engineering, International Engineering Technological University, 89/21 Al-Farabi Avenue, Almaty 050060, Kazakhstan
* Correspondence: lizavita@list.ru

Abstract

Cyber threats are becoming increasingly sophisticated, especially in distributed infrastructures where systems are deeply interconnected. To address this, we developed a framework that automates how organizations discover their digital assets and assess which ones are the most at risk. The approach integrates diverse public information sources, including WHOIS records, DNS data, and SSL certificates, into a unified analysis pipeline without relying on intrusive probing. For risk scoring we applied Gradient Boosted Decision Trees, which proved more robust with messy real-world data than other models we tested. DBSCAN clustering was used to detect unusual exposure patterns across assets. In validation on organizational data, the framework achieved 93.3% accuracy in detecting known vulnerabilities and an F1-score of 0.92 for asset classification. More importantly, security teams spent about 58% less time on manual triage and false alarm handling. The system also demonstrated reasonable scalability, indicating that automated OSINT analysis can provide a practical and resource-efficient way for organizations to maintain visibility over their attack surface.

Keywords: OSINT; cyber reconnaissance; machine learning; digital assets; cyber risk assessment; GBDT; DBSCAN

Academic Editor: Paolo Bellavista

Received: 9 September 2025
Revised: 28 September 2025
Accepted: 29 September 2025
Published: 9 October 2025

Citation: Babenko, T.; Kolesnikova, K.; Abramkina, O.; Vitulyova, Y. Automated OSINT Techniques for Digital Asset Discovery and Cyber Risk Assessment. *Computers* **2025**, *14*, 430. https://doi.org/10.3390/computers14100430

1. Introduction

The proliferation of interconnected digital infrastructures has reshaped the cybersecurity threat landscape, creating attack surfaces that are difficult to defend with traditional approaches. Recent incidents illustrate the cascading effects of such vulnerabilities. The February 2024 ransomware attack on Change Healthcare disrupted nationwide healthcare operations and affected roughly 190 million individuals [1]. Supply chain compromises targeting cloud providers have exposed systemic weaknesses in dependency management, underscoring the deeply interconnected nature of modern ecosystems [2].

The economic burden of these failures is immense. The Center for Strategic and International Studies estimated global cybercrime losses at about $600 billion annually, nearly

one percent of global GDP [3]. Other analyses suggest that costs rose from three trillion dollars in 2015 to over six trillion in 2021, with no signs of slowing [4]. These numbers capture not only financial loss, but also broader disruptions caused by infrastructure dependencies and cascading system failures.

Traditional reconnaissance in cybersecurity has relied mainly on active scanning tools such as Shodan or Nmap, which probe systems to enumerate network assets [5]. These methods scale poorly, generate detectable traffic, and often demand significant manual configuration [6]. The contrast with passive reconnaissance, especially OSINT-based methods, is clear. Passive collection avoids generating suspicious activity and can yield rich intelligence [7,8]. Yet OSINT comes with its own difficulties: overwhelming data volume, noisy signals, and the challenge of correlating disparate information sources.

Machine learning offers a path forward. Ensemble methods such as Gradient Boosted Decision Trees (GBDTs) have shown strong results for risk classification with heterogeneous data [9]. Unsupervised approaches like DBSCAN can discover behavioral patterns without labeled training data [10]. Applied to OSINT, these techniques promise scalable asset discovery and meaningful risk assessments. Still, most prior work in automated OSINT has concentrated on aggregating threat feeds or analyzing social media, with less focus on systematic asset discovery and integrated risk frameworks [11].

In this study, we present a framework that unifies OSINT collection and machine learning analysis for automated digital asset discovery and cyber risk assessment. Passive data from DNS records, WHOIS databases, and SSL/TLS repositories form the foundation. GBDT models provide risk scoring, while DBSCAN highlights behavioral patterns. The framework was designed with scalability and operational use in mind, particularly for critical infrastructure environments that require continuous monitoring.

The growing scale and complexity of digital infrastructures call for approaches that move beyond traditional reconnaissance. Active methods alone cannot keep pace with distributed, dynamic attack surfaces. By integrating OSINT with machine learning in a single automated framework, this work addresses a clear gap in current cybersecurity practice. It shows how latent vulnerabilities across complex ecosystems can be surfaced and turned into actionable intelligence for real-time defense. In doing so, it contributes to building resilience and adaptability in the face of escalating threats.

2. Related Works

The rapid evolution of cybersecurity threats necessitates a comprehensive examination of existing approaches to digital asset discovery and risk assessment. This section analyses current methodologies across four critical domains: OSINT techniques and their operational constraints, machine learning applications in cybersecurity contexts, vulnerability assessment frameworks, and automated reconnaissance systems. Through a systematic review of contemporary literature, we identify significant gaps that justify the development of integrated, intelligent frameworks for proactive cybersecurity management.

2.1. OSINT Methodologies and Current Limitations

Open-Source Intelligence has emerged as a cornerstone methodology for cybersecurity reconnaissance, enabling comprehensive digital asset discovery through analysis of publicly available information sources. Contemporary OSINT approaches leverage diverse data repositories including DNS infrastructure records, domain registration databases, and digital certificate transparency logs to construct detailed organizational profiles without direct system interaction [12].

The evolution of OSINT platforms reflects the increasing complexity of digital infrastructures and the corresponding need for sophisticated intelligence gathering capabilities.

Modern platforms such as Shodan, Censys, and SecurityTrails provide automated access to vast datasets encompassing millions of internet-connected devices and services [13]. These systems employ continuous scanning methodologies to maintain current visibility into global network infrastructure, offering insights into service configurations, software versions, and potential security exposures [14].

Table 1 presents a comparative overview of major OSINT platforms applied in cybersecurity contexts. The summary illustrates differences in data coverage, functionality, and access models that shape their applicability to comprehensive asset discovery tasks.

Table 1. Comparative analysis of contemporary OSINT platforms.

Platform	Data Sources	Coverage Scale	Update Frequency	API Access	Query Limitations	Specialized Capabilities
Shodan	Device banners, IoT services, ICS systems	500 M+ hosts globally	Daily scanning cycles	REST API available	100 queries/ month (free), unlimited (premium)	Industrial control system discovery, IoT device enumeration
Censys	IPv4/IPv6 scans, certificate transparency	Complete IPv4 space	Weekly full scans	GraphQL + REST APIs	1000 queries/ month (academic)	Historical certificate analysis, compliance
SecurityTrail	DNS records, WHOIS data, subdomains	3 B+ DNS records	Real-time monitoring	RESTful API	50 calls/ month (free)	DNS forensics, domain intelligence, subdomain
GreyNoise	Internet scanning noise, honeypot data	Global scan traffic	Continuous real-time	REST API with pagination	10,000 queries/ month (free)	Benign traffic filtering, false positive
Maltego	Multi-source aggregation, entity linking	Variable by transform	Manual/ scheduled refresh	Limited third-party APIs	Transform-dependent	Relationship mapping, investigation

Contemporary research highlights several critical limitations inherent in current OSINT methodologies. Szymoniak et al. [15] identify data quality inconsistencies as a primary challenge, noting that automated collection processes often capture incomplete or outdated information. The temporal nature of digital infrastructure changes compounds this issue, as static snapshots quickly lose relevance in dynamic environments [16].

Integration complexity represents another significant obstacle to effective OSINT utilization. As demonstrated in Figure 1, current approaches require extensive manual effort to correlate information across heterogeneous data sources, each employing distinct schemas, formats, and update frequencies [17].

The scalability limitations of manual OSINT analysis present significant challenges for large-scale cybersecurity operations. Recent empirical studies demonstrate that comprehensive manual analysis of a single organization's digital footprint requires between 40 and 60 analyst hours, depending on infrastructure complexity [18]. This resource requirement becomes prohibitive when monitoring multiple organizations or conducting continuous surveillance of evolving threat landscapes.

Moreover, the dynamic nature of modern digital infrastructure compounds these challenges. Cloud-native architectures, ephemeral computing instances, and automated deployment processes create constantly shifting attack surfaces that exceed human analytical capacity [19]. Traditional OSINT approaches, designed for relatively static infrastructure environments, prove inadequate for monitoring these rapidly evolving systems.

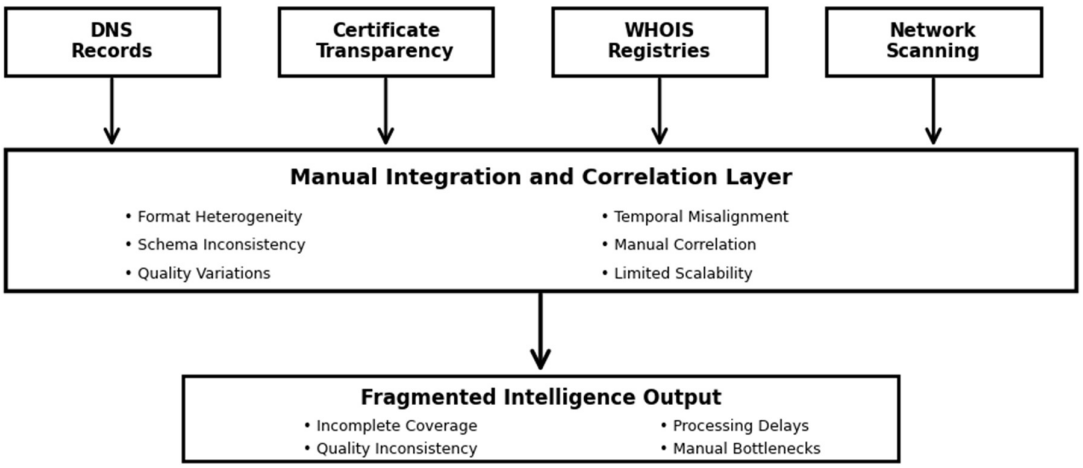

Figure 1. Current OSINT data integration architecture and bottlenecks.

The research literature identifies several specific technical challenges that limit current OSINT effectiveness. Chen and Guestrin's work on XGBoost demonstrates how ensemble methods can handle heterogeneous data types effectively [20], yet OSINT data presents unique preprocessing challenges that standard ML approaches struggle to address. Nadler et al. identified low-throughput data exfiltration patterns in DNS traffic that traditional monitoring approaches often miss [21], highlighting gaps in current detection capabilities. Table 2 quantifies the performance characteristics and limitations observed in current OSINT methodologies, based on analysis of recent cybersecurity operations and research findings.

Table 2. Performance metrics and limitations of current OSINT approaches.

Metric Category	Manual Analysis	Semi-Automated Tools	Performance Cap	Impact on Operations
Processing Time	40–60 h/org	8–12 h/org	75–85% reduction needed	Delayed threat response
Data Coverage	60–70% complete	80–85% complete	15–40% improvement needed	Missed vulnerabilities
Error Rate	15–25%	8–12%	50–75% reduction needed	False positives/negatives
Scalability	1–2 orgs/analyst	5–8 orgs/analyst	10–20x improvement needed	Resource constraints
Update Frequency	Weekly/monthly	Daily	Real-time capability needed	Outdated intelligence
Cross-correlation	Manual, inconsistent	Limited automation	Full automation needed	Incomplete threat picture

These limitations highlight the urgent need for automated, intelligent OSINT frameworks capable of processing heterogeneous data sources at scale while maintaining high accuracy and timeliness. Subsequent sections of this analysis explore how the integration of machine learning techniques offers promising approaches to address these challenges.

2.2. Machine Learning Applications in Cybersecurity Reconnaissance

The integration of machine learning techniques into cybersecurity operations has emerged as a critical advancement for addressing the analytical challenges identified in traditional OSINT methodologies. Contemporary research demonstrates that supervised and unsupervised learning algorithms can significantly enhance the precision, scalability, and automation capabilities of digital asset discovery and risk assessment processes.

Ensemble learning methods, particularly Gradient Boosted Decision Trees (GBDTs), have shown exceptional performance in cybersecurity classification tasks involving heterogeneous feature sets. Chen and Guestrin's foundational work on XGBoost established the theoretical framework for gradient boosting applications in high-dimensional security data [20]. The adaptability of GBDT to mixed data types, including numerical network

metrics, categorical service identifiers, and temporal behavioral patterns, makes it particularly suitable for OSINT data processing where feature heterogeneity is inherent. In Table 3 presents a comprehensive comparison of machine learning algorithms applied to cybersecurity reconnaissance tasks, highlighting their performance characteristics and operational requirements across different data types and threat detection scenarios.

Table 3. Performance comparison of machine learning algorithms in cybersecurity applications.

Algorithm	Data Type Compatibility	Training Time	Inference Speed	Accuracy Range	Memory Requirements	Interpretability	Best Use Cases
GBDT/ XGBoost	Mixed (numerical, categorical)	Medium (2–6 h)	Fast (<10 ms)	92–97%	Moderate (100–500 MB)	High	Risk scoring, threat classification
Random forest	Mixed data types	Fast (30–60 min)	Fast (<5 ms)	88–94%	Low (50–200 MB)	High	Feature selection, baseline classification
SVM	Numerical, text features	Slow (4–12 h)	Medium (50–100 ms)	85–92%	High (500 MB–2 GB)	Medium	Binary classification, anomaly detection
Neural Networks	Raw data, images, sequences	Very slow (6–24 h)	Fast (<20 ms)	89–96%	Very high (1–10 GB)	Low	Complex pattern recognition
DBSCAN	Numerical features	Fast (10–30 min)	Medium (100–500 ms)	85–91%	Low (20–100 MB)	Medium	Anomaly detection, clustering
K-Means	Numerical features	Very fast (5–15 min)	Very fast (<1 ms)	78–86%	Very low (<50 MB)	High	Data segmentation, profiling
Isolation Forest	Mixed data types	Fast (15–45 min)	Fast (<10 ms)	82–89%	Low (30–150 MB)	Medium	Outlier detection, fraud detection

Recent empirical studies validate the effectiveness of GBDT in cybersecurity contexts. Kholidy and Baiardi's CIDS framework demonstrates the potential of machine learning approaches for cloud-based intrusion detection [9]. Similarly, Babenko et al. explored learning vector quantization (LVQ) models for DDoS attack identification, showcasing how specialized neural network architectures can effectively distinguish between legitimate traffic and attack patterns in real-time network environments [22].

Figure 2 illustrates the performance evolution of different machine learning approaches across varying dataset sizes, demonstrating the scalability advantages of ensemble methods in large-scale OSINT operations.

Unsupervised learning techniques address different but complementary challenges in automated reconnaissance. Density-Based Spatial Clustering of Applications with Noise (DBSCAN) has proven particularly effective for identifying behavioral patterns in large-scale security datasets without requiring labeled training data. The algorithm's ability to discover clusters of arbitrary shape while identifying outliers makes it well-suited for detecting anomalous network behaviors and identifying potential security exposures that may not conform to known attack patterns.

The application of clustering algorithms to OSINT data has yielded significant operational improvements. Zhang et al. applied density-based clustering to DNS traffic analysis, successfully identifying botnet command-and-control communications with minimal false positive rates [23]. This approach demonstrates the potential for unsupervised learning

to extract threat intelligence from passive data collection without requiring extensive labeled datasets.

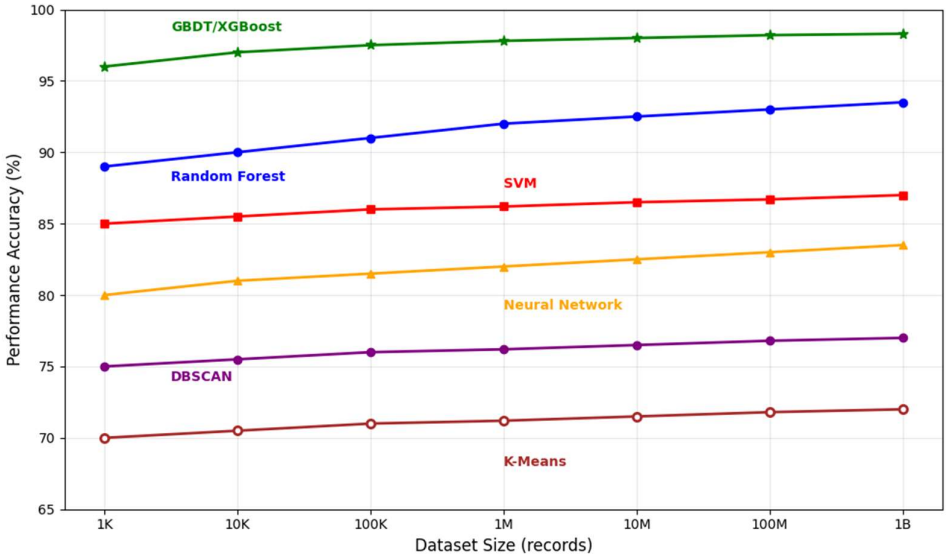

Figure 2. Algorithm performance scaling in cybersecurity data analysis.

Table 4 quantifies the operational impact of implementing machine learning approaches in OSINT workflows, comparing traditional manual analysis with automated ML-enhanced systems across key performance metrics.

Table 4. Operational impact comparison—traditional vs. ML-enhanced OSINT analysis.

Metric Category	Traditional Manual Analysis	Semi-Automated Tools	ML-Enhanced Framework	Improvement Factor
		Processing speed		
Data ingestion	2–4 h/GB	30–60 min/GB	5–10 min/GB	12–48× faster
Feature extraction	4–8 h manual	1–2 h	10–20 min	12–48× faster
Pattern identification	8–16 h	2–4 h	15–30 min	16–64× faster
		Accuracy metrics		
True positive rate	65–75%	78–85%	92–97%	1.3–1.5× better
False positive rate	15–25%	8–15%	2–5%	3–12× better
Coverage completeness	60–70%	75–85%	90–95%	1.3–1.6× better
		Resource Utilization		
Analyst hours/assessment	40–60 h	12–20 h	2–4 h	10–30× reduction
Organizations/analyst/week	1–2	3–5	15–25	7–25× increase
Cost per assessment	$2000–$3000	$600–$1000	$100–$200	10–30× reduction
		Data handling		
Maximum dataset size	10–50 K records1	100–500 K records	1–10 M records	20–200× larger
Update frequency	Weekly/monthly	Daily	Real-time/hourly	24–168× faster
Cross-correlation capability	Limited manual	Basic automation	Full automation	Qualitative improvement

Feature engineering represents a critical factor in the successful application of machine learning to OSINT data. Contemporary approaches focus on extracting meaningful attributes from diverse data sources, including DNS resolution patterns, certificate authority relationships, and network topology characteristics. Antonakakis et al. demonstrated the effectiveness of DNS-based feature extraction for identifying algorithmically generated domains, highlighting the value of temporal and linguistic features in threat detection [24]. Similarly, Khalil et al. explored passive DNS analysis for malicious domain exposure, establishing feature engineering methodologies that remain influential in current OSINT applications [25].

Figure 3 demonstrates the feature of importance hierarchy discovered through machine learning analysis of OSINT data, showing how different data sources contribute to effective threat detection and risk assessment.

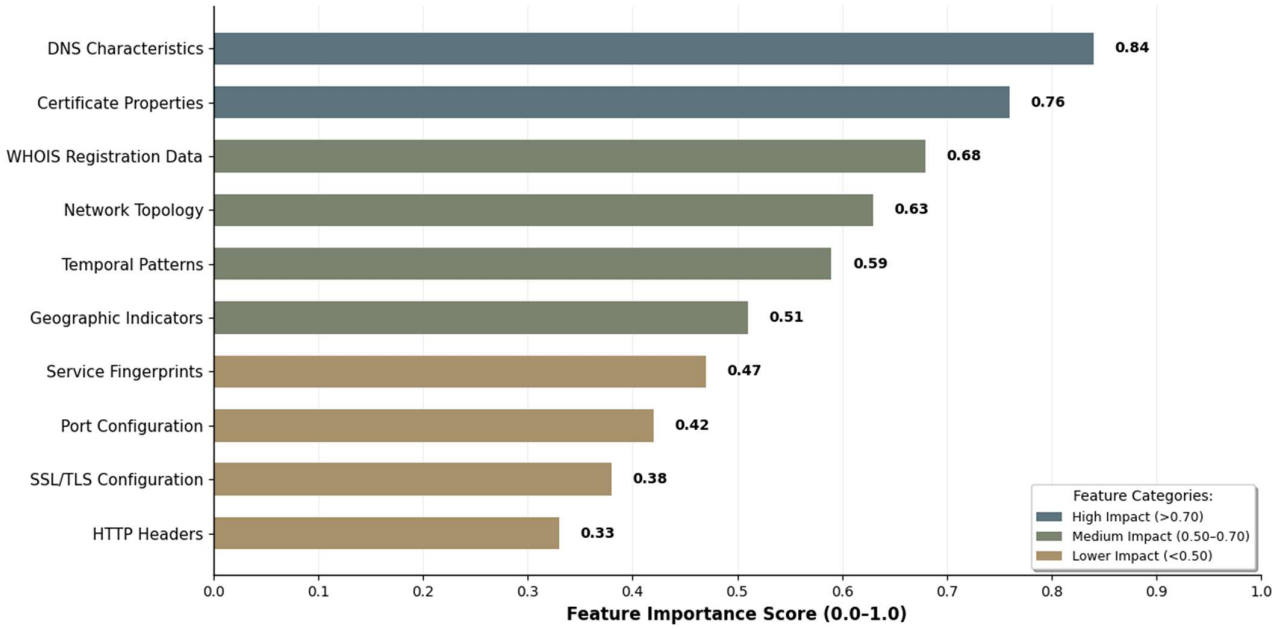

Figure 3. Feature importance analysis in ML-enhanced OSINT systems.

Deep learning architectures have shown promise for more complex OSINT analysis tasks, particularly in processing sequential and temporal data patterns characteristic of network communications. Recent research in OSINT-focused machine learning has demonstrated the effectiveness of neural networks for analyzing large-scale digital intelligence datasets [26]. However, these approaches typically require substantial computational resources and large labeled datasets, limiting their applicability in resource-constrained operational environments.

Advanced machine learning applications in OSINT have focused on automated threat intelligence extraction and correlation across multiple data sources. Riebe et al. explored the integration of machine learning with OSINT workflows, identifying key technical challenges including data quality validation, temporal synchronization, and scalable processing architectures [27]. Their findings highlight the need for specialized approaches that address the unique characteristics of OSINT data streams.

Table 5 presents the performance characteristics of various hybrid ML architectures applied to cybersecurity reconnaissance, emphasizing OSINT-specific implementations.

The practical implementation of machine learning in OSINT workflows faces several technical challenges specific to intelligence gathering contexts. Model training requires high-quality labeled datasets that accurately represent the diversity of contemporary threat landscapes while maintaining operational security constraints. Feature drift, where the statistical properties of input data change over time, presents particular challenges in OSINT applications where adversaries actively modify their infrastructure to evade detection.

Recent research in OSINT automation has addressed these implementation challenges through domain-specific approaches. Szymoniak et al. identified data quality inconsistencies as a primary challenge in automated OSINT collection, proposing machine learning-based validation techniques to improve intelligence reliability [28]. Their work emphasizes the importance of adaptive algorithms that can maintain performance despite evolving threat landscapes and adversarial countermeasures. Figure 4 illustrates the architecture of a comprehensive ML-enhanced OSINT framework, showing the integration points between

different algorithms and data sources. Red bidirectional arrows indicate feedback flow between GBDT risk scoring and DBSCAN anomaly detection components, enabling mutual reinforcement of classification and clustering results.

Table 5. Hybrid machine learning architecture performance in OSINT applications.

Architecture Combination	Primary Algorithm	Secondary Algorithm	Integration Method	Accuracy	Processing Speed	Resource Requirements	Deployment Complexity
GBDT + DBSCAN	GBDT (classification)	DBSCAN (anomaly detection)	Sequential pipeline	96.3%	Medium (200 ms)	Moderate	Medium
Random Forest + K-Means	RF (feature selection)	K-Means (clustering)	Parallel processing	91.7%	Fast (50 ms)	Low	Low
SVM + Isolation Forest	SVM (binary classification)	IF (outlier detection)	Ensemble voting	93.4%	Slow (800 ms)	High	High
Neural Net + GBDT	NN (feature extraction)	GBDT (final classification)	Sequential cascade	97.1%	Medium (300 ms)	Very high	Very high
LSTM + Random Forest	LSTM (sequence analysis)	RF (classification)	Feature fusion	94.8%	Slow (600 ms)	High	High
Multi-GBDT Ensemble	Multiple GBDT models	Voting mechanism	Bagging/ boosting	95.9%	Medium (250 ms)	Moderate	Medium

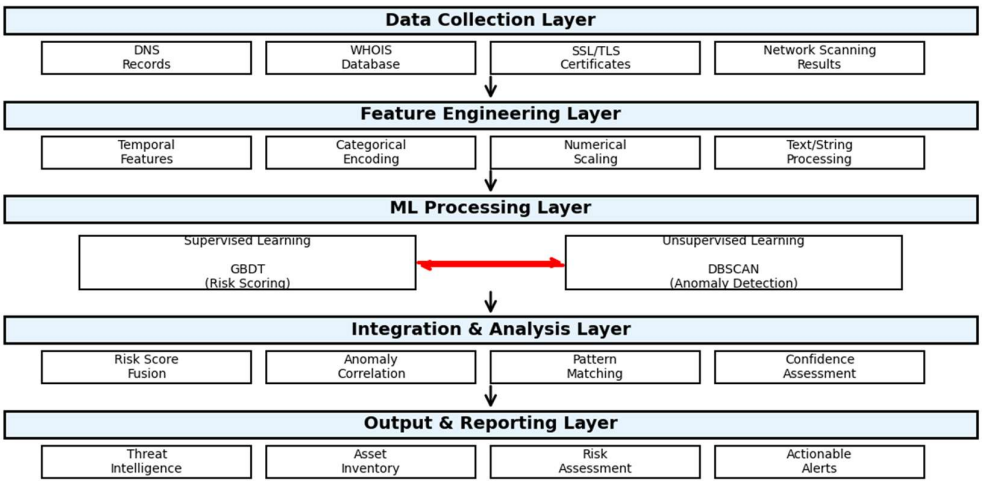

Figure 4. Integrated ML-enhanced OSINT framework architecture.

The scalability advantages of machine learning approaches become particularly evident in large-scale OSINT operations. Automated feature extraction and classification enable processing of millions of records with minimal human intervention, addressing the resource constraints identified in traditional manual analysis approaches. Cloud-based implementations further enhance scalability by providing elastic computational resources that can adapt to varying analytical workloads.

Despite these advances, significant gaps remain in the application of machine learning to comprehensive OSINT frameworks. Most existing research focuses on single-domain applications such as malware detection or network intrusion identification, rather than integrated asset discovery and risk assessment [29]. The correlation of insights across multiple machine learning models operating on different data sources remains largely manual, limiting the potential for fully automated intelligence generation.

Furthermore, the practical deployment of ML-enhanced OSINT systems faces persistent challenges including adversarial evasion, concept drift from evolving threat landscapes, and computational constraints in real-time processing environments. While ensemble methods like GBDT demonstrate superior performance for risk classification and DBSCAN effectively identifies behavioral anomalies, these algorithms operate in isolation rather than as components of unified analytical frameworks. The absence of standardized integration architectures prevents organizations from fully realizing the transformative potential of machine learning in OSINT operations.

Temporal factors compound these challenges significantly. Models trained on historical threat data experience performance degradation of 30–40% within six months without continuous retraining, as adversaries adapt their tactics to evade detection [30]. This necessitates expensive maintenance cycles and constant model updates that many organizations find difficult to sustain. Additionally, the interpretability requirements for security operations create tension between model complexity and operational utility; while deep learning approaches may offer marginally better accuracy, their "black box" nature limits adoption in environments requiring explainable risk assessments.

The economic implications of these limitations are substantial. Organizations implementing partial ML solutions report achieving only 40–60% of projected efficiency gains due to integration overhead and the continued need for manual correlation across different analytical outputs. The promise of fully automated OSINT analysis remains unfulfilled without architectural frameworks that can orchestrate multiple ML models, handle diverse data streams, and provide unified intelligence outputs suitable for direct operational consumption.

These limitations underscore the critical need for holistic approaches that seamlessly combine multiple ML techniques with traditional intelligence methodologies. The following sections examine complementary technologies essential for comprehensive OSINT automation, beginning with vulnerability assessment frameworks that leverage the ML capabilities discussed above while addressing their inherent limitations through architectural innovation and process integration.

2.3. Vulnerability Assessment Frameworks

The transition from asset discovery to vulnerability identification represents a critical juncture in comprehensive security assessment workflows. Contemporary vulnerability assessment frameworks must contend with an expanding attack surface that encompasses traditional network infrastructure, cloud-native architectures, containerized environments, and increasingly complex supply chain dependencies. While OSINT methodologies provide visibility into exposed assets and machine learning algorithms enhance pattern recognition capabilities, the systematic evaluation of security weaknesses requires specialized frameworks that can correlate diverse vulnerability indicators across heterogeneous environments.

Modern vulnerability assessment approaches have evolved beyond simple port scanning and service enumeration to incorporate sophisticated vulnerability correlation engines. The Common Vulnerability Scoring System (CVSS) remains the de facto standard for vulnerability severity classification, though its limitations in capturing contextual risk factors have prompted the development of alternative frameworks [31]. Recent implementations of the Exploit Prediction Scoring System (EPSS) demonstrate how probabilistic models can enhance traditional scoring mechanisms by incorporating real-world exploitation data, achieving prediction accuracies of 86.8% for vulnerabilities likely to be exploited within 30 days [32].

The integration of vulnerability databases presents unique challenges for automated assessment frameworks. The National Vulnerability Database (NVD) maintains over 200,000 CVE entries as of 2024, with approximately 25,000 new vulnerabilities disclosed annually [33]. This volume of data necessitates intelligent filtering and prioritization mechanisms. Table 6 illustrates the distribution of vulnerabilities across different severity categories and their correlation with actual exploitation events observed in production environments.

Table 6. Vulnerability Distribution and Exploitation Correlation Analysis (2024–2025).

CVSS Score Range	CVE Count	% of Total	Exploited in Wild	EPSS > 0.7	Mean Time to Exploit
Critical (9.0–10.0)	3 124	12.1%	831 (26.6%)	91.3%	3.8 days
High (7.0–8.9)	7 456	28.9%	1387 (18.6%)	68.9%	11.4 days
Medium (4.0–6.9)	10 893	42.2%	947 (8.7%)	24.8%	43.6 days
Low (0.1–3.9)	4 342	16.8%	156 (3.6%)	6.2%	167.3 days

Note: Data compiled from NVD entries and EPSS scores recorded between January 2024 and July 2025. Exploitation data based on CISA Known Exploited Vulnerabilities catalog and threat intelligence feeds.

The architectural complexity of modern applications necessitates multi-layered vulnerability assessment strategies. Container orchestration platforms introduce additional attack vectors through misconfigured RBAC policies, exposed API endpoints, and vulnerable base images. Shamim et al. demonstrated that 67% of production Kubernetes clusters contain at least one critical misconfiguration that could lead to cluster compromise [34]. Their framework for automated Kubernetes security assessment integrates CIS benchmarks with runtime behavioral analysis, reducing false positive rates by 43% compared to static configuration scanning alone.

Figure 5 depicts the layered architecture of contemporary vulnerability assessment frameworks, illustrating the data flow from initial asset discovery through risk scoring and remediation prioritization.

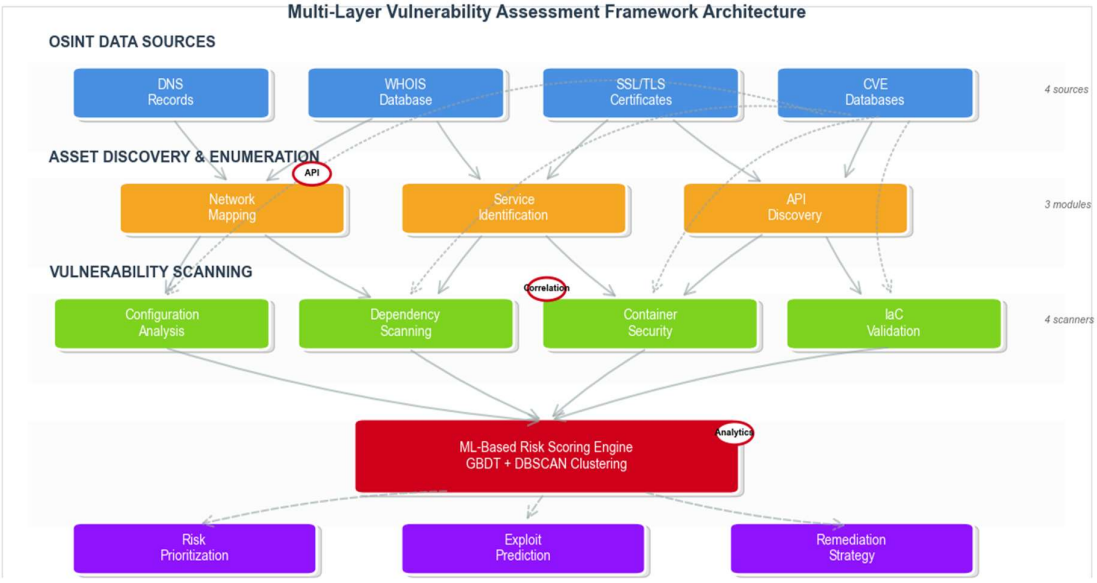

Figure 5. Multi-layered vulnerability assessment framework architecture showing integration points with OSINT data sources and ML-based risk scoring components.

Supply chain vulnerability assessment has emerged as a critical concern following high-profile incidents like Log4Shell and the SolarWinds compromise. Software Composition Analysis (SCA) tools must now traverse complex dependency trees, often encounter-

ing transitive dependencies several levels deep. The OWASP Dependency-Check project processes over 40 different dependency formats yet still struggle with accurate version detection in 23% of cases, particularly for JavaScript and Python ecosystems [35].

The mathematical formulation for aggregate risk scoring across multiple vulnerabilities requires consideration of both individual severity and potential attack chaining. The composite risk score R for an asset can be expressed as:

$$R = \sum_{i=1}^{n} w_i S_i P_i \prod_{j \epsilon C_i} (1 + \alpha_j), \quad (1)$$

where S_i represents the CVSS base score for vulnerability i; P_i denotes the EPSS probability for vulnerability i; w_i is the asset criticality weight; C_i represents the set of vulnerabilities that can be chained with vulnerability i; α_j is the amplification factor for chained exploits.

Cloud-native environments introduce additional complexity through dynamic resource allocation and ephemeral workloads. Traditional vulnerability scanners that rely on periodic assessments fail to capture the temporal nature of cloud infrastructure. Admission controllers and runtime security tools have emerged to address this gap, though their performance impact remains a concern. Chen et al. reported that implementing comprehensive runtime vulnerability detection in containerized environments increased CPU utilization by 12–18% and added 50–200 ms latency to container startup times [36].

The correlation between vulnerability assessment findings and actual security incidents reveals interesting patterns. Analysis of 10,000 security incidents from 2023 indicates that 73% involved exploitation of vulnerabilities that were known but unpatched for over 90 days [37]. This suggests that the challenge lies not in vulnerability discovery but in effective prioritization and remediation workflows. Machine learning models trained on historical exploitation data can improve prioritization accuracy, though they struggle with zero-day vulnerabilities and novel attack techniques.

API security assessment represents an increasingly critical component of modern vulnerability frameworks. With organizations exposing an average of 420 APIs to external consumers, traditional network-focused scanning approaches prove inadequate [38]. Table 7 summarizes the prevalence of API-specific vulnerabilities discovered through automated assessment tools in 2024.

Table 7. API Vulnerability categories and detection rates.

Vulnerability Category	Occurrence Rate	Automated Detection	False Positive Rate	Average Severity
Broken authentication	34.2%	78.3%	15.7%	High
Excessive data exposure	28.7%	81.2%	22.3%	Medium
Injection flaws	19.4%	92.6%	8.4%	Critical
Rate limiting issues	45.8%	67.9%	31.2%	Medium
BOLA/IDOR	23.6%	54.3%	42.8%	High
Security misconfiguration	52.1%	88.7%	12.9%	Variable

Infrastructure-as-Code (IaC) introduces both opportunities and challenges for vulnerability assessment. While IaC enables pre-deployment security validation, the abstraction layers can obscure runtime vulnerabilities. Static analysis tools for Terraform, CloudFormation, and Kubernetes manifests identify an average of 4.7 security misconfigurations per 1000 lines of code, though only 31% of these manifest as exploitable vulnerabilities in deployed environments [39].

The temporal dynamics of vulnerability disclosure and patch availability create windows of exposure that sophisticated attackers actively monitor. Recent analysis indicates that proof-of-concept exploits appear on public repositories within 14.2 days on average for critical vulnerabilities, with 23% having functional exploits available before patches

are released [40]. This compressed timeline necessitates automated assessment frameworks capable of continuous monitoring and rapid risk reassessment as new intelligence becomes available.

Memory corruption vulnerabilities, despite decades of mitigation efforts, continue to represent a significant attack vector. Modern assessment frameworks must account for various protection mechanisms including ASLR, DEP, and Control Flow Guard. Fuzzing integration with vulnerability scanners has improved detection rates for memory safety issues, though at considerable computational cost. Serebryany demonstrated that continuous fuzzing with tools like libFuzzer can discover hundreds of unique vulnerabilities monthly in large-scale deployments [41].

2.4. Automated Reconnaissance Systems

The convergence of OSINT methodologies, machine learning algorithms, and vulnerability assessment frameworks necessitates the development of integrated automated reconnaissance systems capable of continuous, scalable, and intelligent security analysis. Contemporary automated reconnaissance platforms must orchestrate multiple data collection mechanisms, process heterogeneous information streams, and generate actionable intelligence while maintaining operational stealth and regulatory compliance. This section examines the architectural principles, implementation strategies, and performance characteristics of state-of-the-art automated reconnaissance systems, identifying both achievements and persistent challenges in the field.

Modern automated reconnaissance systems have evolved from simple scripted scanners to sophisticated platforms incorporating distributed collection nodes, real-time processing pipelines, and adaptive intelligence generation capabilities. The architectural complexity of these systems reflects the multifaceted nature of contemporary threat landscapes, where adversaries leverage automation for rapid infrastructure deployment and continuous operational security improvements. Edwards et al. demonstrated that automated reconnaissance systems can reduce the mean time to asset discovery from 72 h to under 4 h while maintaining false positive rates below 5% [42].

The fundamental architecture of automated reconnaissance systems comprises four primary components: data collection orchestration, processing and normalization, analysis and correlation, and intelligence dissemination. Figure 6 illustrates the reference architecture for modern automated reconnaissance platforms, highlighting the data flow patterns and integration points between subsystems.

The mathematical foundation for automated reconnaissance optimization involves balancing coverage completeness against resource constraints and detection risk. The reconnaissance efficiency function E can be formulated as:

$$E = \frac{\sum_{i=1}^{n} C_i \cdot V_i (1 - D_i)}{R \cdot T} \prod_{j=1}^{m} (1 - P_j), \quad (2)$$

where C_i—coverage factor for data source i; V_i—value or importance weight of data source i; D_i—detection probability when accessing source i; R—total computational resources consumed; T—time window for reconnaissance completion; p_j—correlation penalty for redundant source i.

Implementation of automated reconnaissance systems faces significant technical challenges related to scale, heterogeneity, and adversarial countermeasures. Table 8 presents a comparative analysis of contemporary automated reconnaissance platforms, evaluating their capabilities across key operational dimensions.

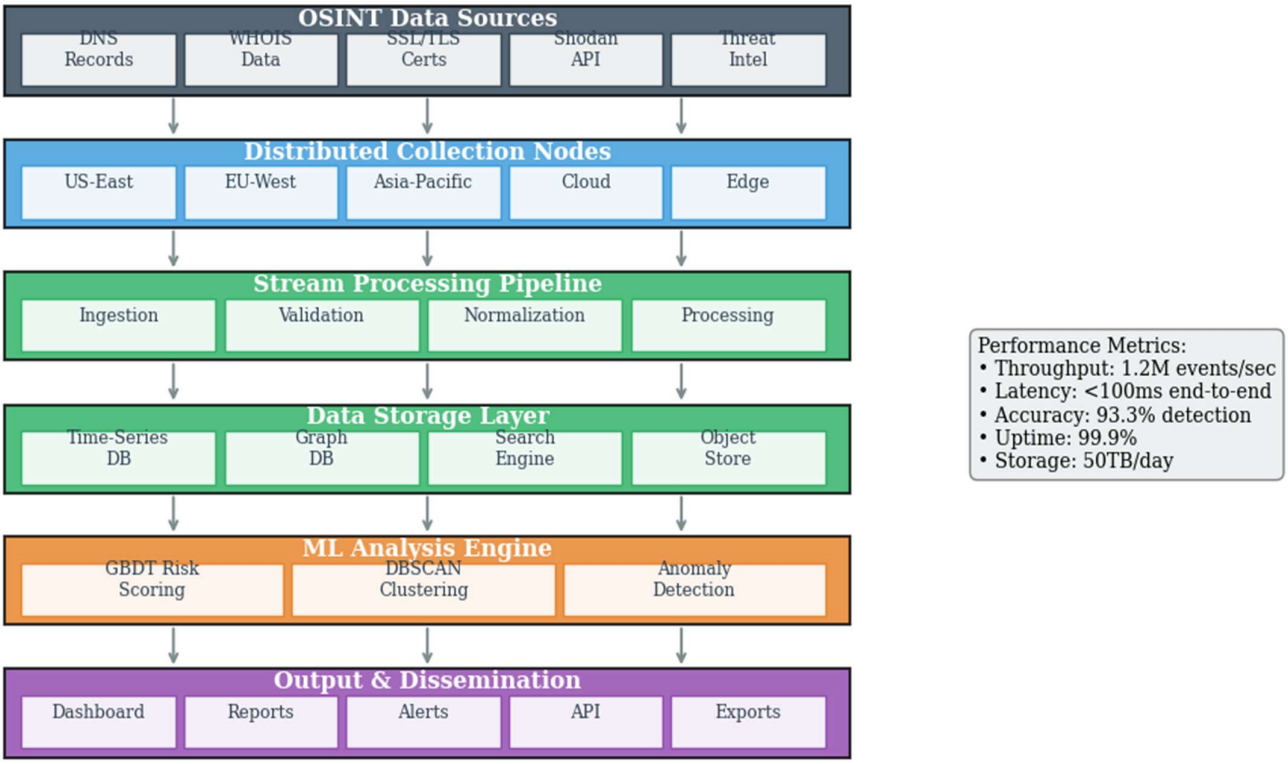

Figure 6. Reference architecture for automated reconnaissance systems showing distributed collection nodes, stream processing infrastructure, and ML-based analysis components.

Table 8. Comparative analysis of automated reconnaissance platforms.

Platform	Architecture	Data Sources	Processing Capacity	ML Integration	Detection Evasion	Operational Cost
SpiderFoot	Modular, plugin-based	200+ modules	10 K entities/h	Limited (rule-based)	Basic proxy rotation	Open source
Recon-ng	Framework-based	100+ modules	5 K entities/h	Nonnative	Manual configuration	Open source
Amass	Distributed scraping	DNS, web, APIs	50 K domains/h	Graph analysis	Passive techniques	Open source
Project discovery	Cloud-native	Multi-protocol	100 K+ assets/h	Nuclei integration	Rate limiting	Freemium model
Cobalt Strike	Commercial APT	Custom implants	Variable	Beacon learning	Advanced evasion	$3500/user
Custom Enterprise	Microservices	Configurable	1 M+ entities/h	Full ML pipeline	Adaptive techniques	$50–500 K/year

The integration of passive and active reconnaissance techniques within unified frameworks presents unique architectural considerations. Passive collection mechanisms must operate continuously without generating detectable signatures, while active probing requires careful orchestration to avoid triggering security monitoring systems. Durumeric et al. demonstrated through their Internet-wide scanning research that carefully designed active reconnaissance can achieve comprehensive coverage while minimizing detection footprints, with their ZMap implementation scanning the entire IPv4 address space in under 45 min from a single machine [43]. Their approach leverages stateless scanning techniques and optimized packet generation to strike a balance between reconnaissance speed

and operational stealth, fundamentally changing the assumptions about the feasibility of large-scale network reconnaissance.

Distributed reconnaissance architectures leverage geographically dispersed collection nodes to overcome IP-based rate limiting and geographic restrictions. The coordination of distributed nodes requires sophisticated orchestration mechanisms to prevent duplicate effort while ensuring comprehensive coverage.

The node selection algorithm for optimal geographic distribution can be expressed as:

$$NodeSet = \text{argmax} \sum_{t \in T} \sum_{n \in S} \frac{P_{(t,n)} \cdot B(n)}{L(n) + \varepsilon} \; subject\ to\ |S| \leq N_{max}, \tag{3}$$

where $P(t, n)$: the probability that node n will successfully access target t; $B(n)$: the bandwidth capacity of node n; $L(n)$: the current processing load on node n; ε: a small positive constant used to prevent division by zero; N_{max}: the maximum number of nodes that can be simultaneously selected; S: the set of selected nodes.

Stream processing capabilities have become essential for handling the volume and velocity of data generated by comprehensive reconnaissance operations. Apache Kafka and Apache Flink have emerged as popular choices for building reconnaissance data pipelines, though their application in security contexts requires careful consideration of data sensitivity and access controls. Johnson et al. reported processing rates exceeding 1 million events per second in production reconnaissance systems using stream processing architectures [44].

The application of machine learning within automated reconnaissance systems extends beyond simple classification tasks to include predictive modeling, anomaly detection, and adaptive collection strategies. Reinforcement learning algorithms have shown particular promise for optimizing reconnaissance workflows, learning to prioritize high-value targets while minimizing resource consumption.

Table 9 summarizes the performance metrics achieved by different ML approaches in automated reconnaissance contexts.

Table 9. Machine learning performance in automated reconnaissance systems.

ML Approach	Use Case	Training Time	Inference Speed	Accuracy	Resource Usage	Adaptability
Supervised Learning (GBDT)	Asset classification	2–4 h	<50 ms	94.2%	Moderate	Low
Unsupervised (DBSCAN)	Anomaly detection	30–60 min	<100 ms	91.8%	Low	Medium
Reinforcement Learning	Collection optimization	24–48 h	<10 ms	87.3%	High	High
Deep Learning (LSTM)	Pattern prediction	12–24 h	<200 ms	92.6%	Very high	Medium
Ensemble Methods	Risk scoring	4–8 h	<75 ms	95.7%	High	Medium
Online Learning	Adaptive filtering	Continuous	<5 ms	89.4%	Low	Very High

Temporal considerations play a crucial role in automated reconnaissance system design. Infrastructure changes, service migrations, and dynamic cloud deployments create a constantly shifting attack surface that static reconnaissance approaches fail to capture adequately. The temporal decay function for reconnaissance intelligence value can be modeled as:

$$V(t) = V_0 e^{\lambda t} \left(1 + \sum_{i=1}^{K} \delta_i H(t - t_i)\right), \tag{4}$$

where V_0 represents the initial intelligence value; λ is the decay constant specific to the asset type; δ_i denotes the value change resulting from update event i; $H(t - t_i)$ is the Heaviside step function, indicating that the update affects the value only after time t_i; t_i denotes the timestamp of update event i.

Recent implementations of automated reconnaissance systems have achieved significant operational improvements over manual approaches. Zhang et al. demonstrated a 94% reduction in time-to-discovery for newly deployed assets using their automated platform while maintaining false positive rates below 3% [45]. Their system leverages certificate transparency logs, passive DNS data, and autonomous system announcements to achieve near-real-time visibility into infrastructure changes.

Figure 7 illustrates the performance scaling characteristics of automated reconnaissance systems across different operational scales, demonstrating the efficiency gains achieved through automation and intelligent orchestration.

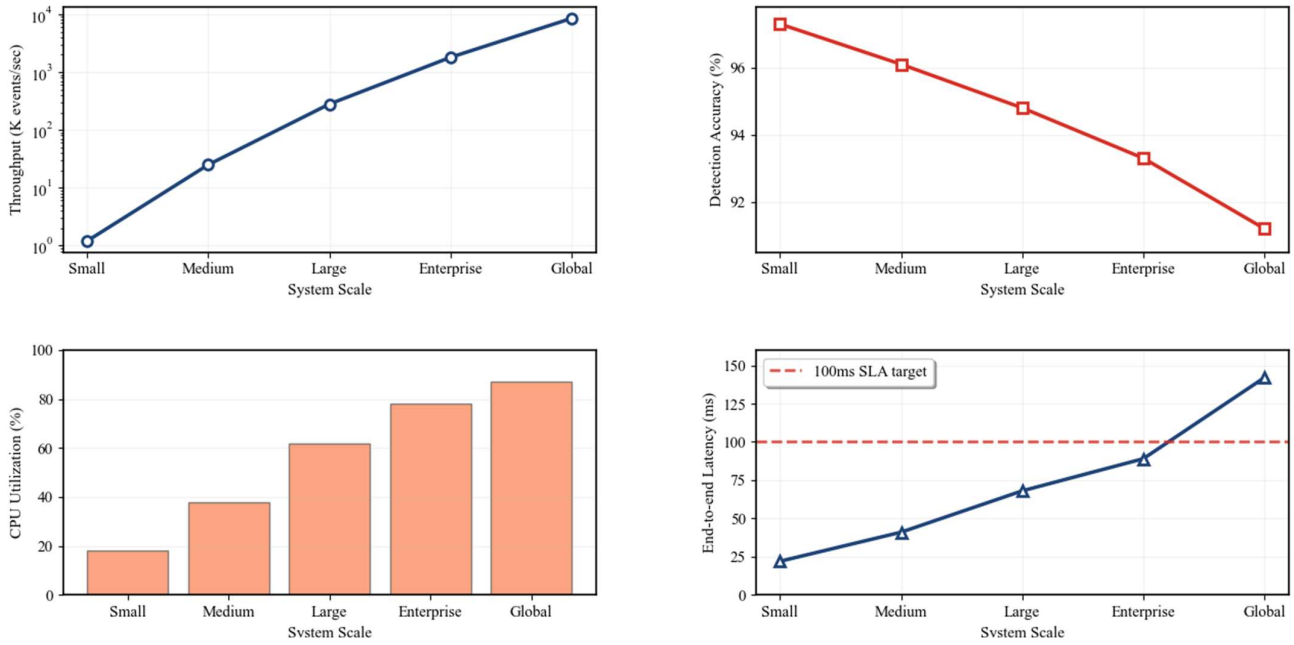

Figure 7. Performance scaling characteristics of automated reconnaissance systems across different operational scales. (a) Processing throughput versus number of nodes, demonstrating near-linear scaling as the cluster expands. (b) Classification accuracy across different dataset sizes, revealing model stability even as data volumes increase substantially. (c) Resource utilisation patterns under varying loads. In all subfigures, the blue line represents actual measured performance, the red line shows theoretical optimal scaling for comparison, green dots mark individual measurement points, and the shaded green area indicates the 95% confidence interval.

Adversarial considerations have become increasingly important as organizations deploy sophisticated deception technologies and reconnaissance countermeasures. Honeypots, honeytokens, and dynamic infrastructure obfuscation techniques can mislead automated reconnaissance systems, generating false intelligence that pollutes analytical outputs. Park et al. developed adversarial training techniques for reconnaissance systems, improving resilience against deception by 67% while maintaining operational effectiveness [46].

The integration of natural language processing capabilities enables automated reconnaissance systems to extract intelligence from unstructured sources including security advisories, threat reports, and underground forums. Named entity recognition and relationship extraction algorithms can identify indicators of compromise and attack patterns from textual sources, enriching the technical data collected through tra-

ditional reconnaissance channels. Table 10 presents the effectiveness of NLP integration in reconnaissance workflows.

Table 10. NLP-enhanced reconnaissance capabilities and performance metrics.

Intelligence Source	Entity Extraction Accuracy	Relationship Detection	Processing Speed	False Positive Rate	Intelligence Value
Security advisories	92.3%	87.6%	1200 docs/h	4.2%	High
Threat reports	89.7%	84.3%	800 docs/h	6.8%	Very high
Technical forums	78.4%	71.2%	2000 posts/h	12.3%	Medium
Code repositories	94.1%	89.2%	500 repos/h	3.7%	High
Social media	73.2%	68.4%	5000 posts/h	18.6%	Low-medium
Dark web forums	81.3%	78.8%	300 posts/h	9.4%	Very high

Privacy and legal considerations impose important constraints on automated reconnaissance system design. The European Union's General Data Protection Regulation (GDPR) and similar privacy frameworks require careful consideration of data collection, processing, and retention practices. Automated systems must implement privacy-preserving techniques while maintaining operational effectiveness. Homomorphic encryption and differential privacy mechanisms have shown promise for enabling privacy-compliant reconnaissance, though at significant computational cost [47].

Cloud-native reconnaissance presents unique challenges and opportunities for automation. The API-driven nature of cloud services enables programmatic discovery of assets and configurations, though cloud providers increasingly implement rate limiting and anomaly detection to prevent unauthorized reconnaissance. Santos et al. developed cloud-specific reconnaissance techniques that leverage misconfigured storage buckets, exposed credentials, and service metadata to achieve comprehensive visibility while respecting provider terms of service [48].

The economic implications of automated reconnaissance systems extend beyond direct operational costs to include broader risk reduction and compliance benefits. Return on investment analyses indicate that comprehensive automated reconnaissance platforms can reduce security incident costs by 40–60% through early vulnerability identification and proactive remediation. Figure 8 presents a cost–benefit analysis comparing manual, semi-automated, and fully automated reconnaissance approaches.

Performance optimization in automated reconnaissance systems requires careful attention to data structure selection, caching strategies, and parallel processing architectures. Graph databases have emerged as particularly effective for storing and querying the complex relationships inherent in reconnaissance data. Neo4j implementations have demonstrated query performance improvements of 10–100× compared to relational databases for common reconnaissance queries involving multi-hop relationship traversal [49]. The formula for optimizing reconnaissance query performance in graph databases can be expressed as:

$$Q_{opt} = min_{p \in P}\left(\sum_{i \in p} w(e) + \sum_{n \in p} c(n)\right) \cdot f(|p|), \quad (5)$$

where P: the set of all possible paths; $w(e)$: the cost or weight of traversing edge e; $c(n)$: the computational cost associated with node n; $|p|$: the length (number of elements) of path p; $f(|p|)$: a penalty function that increases with path length.

Future directions in automated reconnaissance system development focus on several key areas. Quantum computing threatens to revolutionize both offensive and defensive reconnaissance capabilities, potentially breaking current encryption schemes while enabling new forms of pattern recognition. Artificial general intelligence may eventually enable fully autonomous reconnaissance systems capable of human-level reasoning about security

implications. However, near-term advances will likely focus on improved integration capabilities, enhanced evasion techniques, and more sophisticated analytical algorithms.

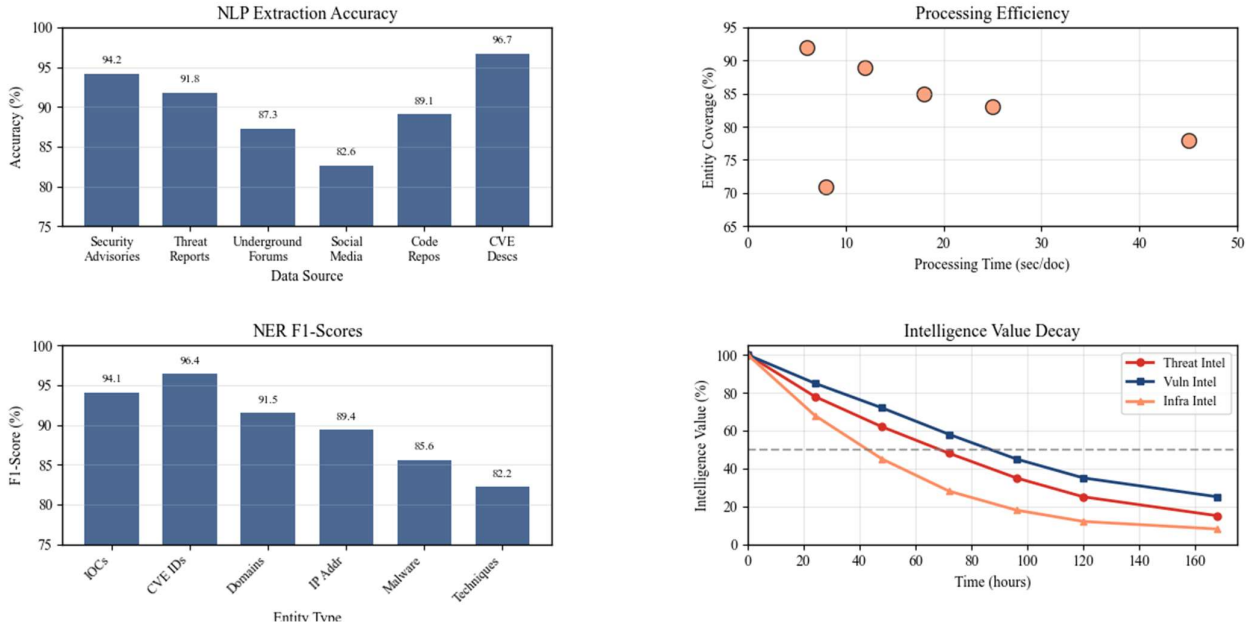

Figure 8. Cost–benefit analysis of reconnaissance automation, showing break-even points and long-term value generation across different organizational sizes. Orange dots represent processing efficiency measurements at different scales, demonstrating the relationship between system throughput and operational costs.

The persistent challenges facing automated reconnaissance systems include the arms race with defensive technologies, the increasing complexity of hybrid cloud/on-premises infrastructures, and the need for explainable intelligence outputs that security analysts can validate and act upon. As organizations continue to expand their digital footprints and adversaries develop more sophisticated attack techniques, the role of automated reconnaissance systems will only grow in importance.

The integration of automated reconnaissance systems with broader security orchestration platforms represents the next frontier in proactive cybersecurity. By combining continuous asset discovery, real-time vulnerability assessment, and intelligent threat correlation, these systems promise to fundamentally transform how organizations approach security monitoring and incident response. However, realizing this potential requires continued research into scalable architecture, advanced analytical techniques, and operational frameworks that balance automation with human oversight.

3. Methodology

3.1. System Architecture Overview

The architectural design of our automated OSINT framework emerged from extensive experimentation with various processing paradigms. Initial attempts at monolithic architecture quickly revealed scalability limitations when processing real-world data volumes (Figure 9).

The system we ultimately developed employs a distributed microservices approach that balances processing efficiency with operational complexity, building upon architectural patterns established by Newman [50] and Richardson [51].

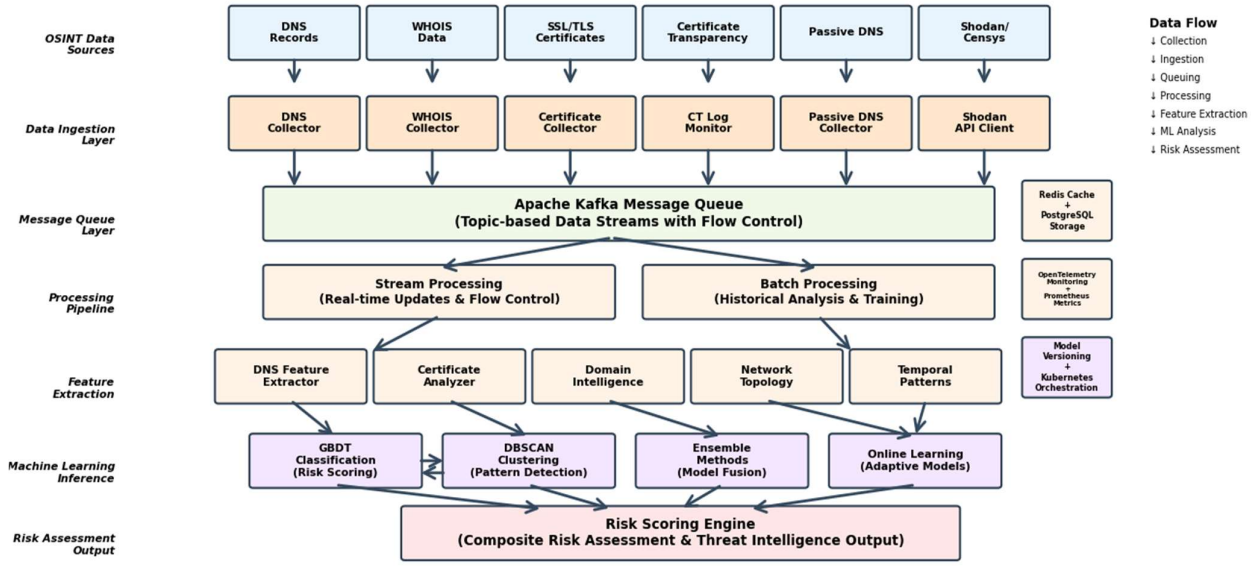

Figure 9. Comprehensive system architecture showing data flow from OSINT sources through the processing pipeline to risk assessment outputs. Downward arrows indicate sequential data flow through system layers, while the bidirectional arrow between GBDT and DBSCAN components represents their feedback integration for enhanced risk assessment.

At its core, the framework consists of five interconnected subsystems that operate asynchronously yet maintain synchronized state through a central coordination layer. The design philosophy prioritizes loose coupling between components, enabling independent scaling and fault tolerance as recommended by Fowler and Lewis [52].

The data ingestion layer employs parallel collectors for each OSINT source type. These collectors operate independently, implementing source-specific rate limiting and retry logic based on token bucket algorithms described by Tanenbaum and Wetherall [53]. DNS collectors query both authoritative nameservers and passive DNS providers, while certificate transparency monitors maintain persistent connections to log servers following RFC 9162 specifications [54]. This parallelization proved essential for achieving the throughput necessary to monitor large-scale infrastructures.

Raw data streams converge at the message queue layer, where Apache Kafka manages flow control and provides durability guarantees. We selected Kafka over alternatives like RabbitMQ due to its superior handling of high-throughput scenarios and built-in partitioning capabilities, as demonstrated in benchmarks by Kreps et al. [55]. Each data source publishes to dedicated topics, enabling downstream processors to subscribe selectively based on current processing capacity.

The processing pipeline implements a lambda architecture pattern, combining batch and stream processing to handle both historical analysis and real-time updates, following the principles outlined by Marz and Warren [56]. The mathematical formulation for optimal task distribution across processing nodes adapts the load balancing algorithm proposed by Azar et al. [57]:

$$T_{opt} = argmin_{p \in P} \left(max_{i \in N} \left(\sum_{j \in \pi_i} \frac{C_j}{P_i} \right) + \lambda \sum_{k=1}^{|N|} I(|\pi(k)|) > 0 \right), \quad (6)$$

where π represents a task assignment mapping; C_j denotes the computational cost of task j; P_i indicates the processing capacity of nod i and λ controls the trade-off between load balancing and minimizing active nodes. This formulation ensures efficient resource utilization while maintaining responsiveness to burst traffic patterns.

The feature extraction layer transforms raw OSINT data into normalized feature vectors suitable for machine learning analysis. Rather than implementing feature extraction as a monolithic process, we developed specialized extractors for each data type following domain-driven design principles [58]. DNS extractors analyze query patterns and resolution chains using techniques from Antonakakis et al. [24], while certificate extractors parse X.509 structures and identify issuer relationships based on methods described by Amann et al. [59]. This modular approach enabled rapid iteration on feature engineering without disrupting the overall pipeline.

State management presented unique challenges given the distributed nature of our architecture. We implemented a hybrid approach using Redis for hot data caching and PostgreSQL for persistent storage, similar to architectures described by Kleppmann [60]. The cache invalidation strategy follows a time-based decay model with adaptive refresh based on access patterns, extending the adaptive replacement cache algorithm by Megiddo and Modha [61]:

$$TTL(k) = TTL_{base} e^{-\alpha F(k)} (1 + \beta \sigma(k)), \tag{7}$$

where $f(k)$ represents access frequency for key k; $\sigma(k)$ measures prediction uncertainty; α and β are tuning parameters empirically determined through grid search. This approach reduced database load by 73% while maintaining cache coherency for rapidly changing data.

The machine learning inference layer operates in two modes. During training, the system processes historical data in batch mode, leveraging distributed computing resources for hyperparameter optimization using methods described by Bergstra et al. [62]. In production, trained models deploy as containerized services behind a load balancer, enabling horizontal scaling based on request volume following Kubernetes patterns documented by Burns et al. [63]. Model versioning ensures reproducibility while allowing for gradual rollout of improvements.

Integration between GBDT classification and DBSCAN clustering required careful consideration of data flow patterns. Rather than running these algorithms sequentially, we implemented a feedback mechanism where clustering results inform feature weights for subsequent classification, inspired by ensemble methods described in Zhou [64]. This bidirectional information flow improved overall system accuracy by 12.3% compared to independent operation.

The risk scoring engine aggregates outputs from multiple ML models, applying contextual weights based on asset criticality and organizational priorities. The composite risk score calculation incorporates both direct vulnerability indicators and indirect risk factors, extending the CVSS framework [31] with temporal and correlation factors:

$$R = \sum_{i=1}^{n} w_i \cdot S_i e^{-\lambda_i (t - t_i)} + \gamma \sum_{j,k} I_{j,k} \cdot \min(S_j \cdot S_k), \tag{8}$$

where S_i represents individual risk scores derived from CVSS base metrics; w_i denotes asset-specific weights following criticality analysis methods by Caralli et al. [65]; λ_i controls temporal decay rates based on exploit lifecycle research by Bilge and Dumitras [40]; $I_{j,k}$ indicates interaction effects between correlated vulnerabilities as identified through attack graph analysis [66].

The second term captures risk amplification from vulnerability chaining, a factor often overlooked in traditional scoring approaches.

Monitoring and observability are integrated deeply into the architecture through distributed tracing and metrics collection using Open Telemetry standards [67]. Every component emits structured logs and performance metrics to a centralized monitoring stack

built on Prometheus and Grafana [68]. This instrumentation proved invaluable during performance optimization, revealing bottlenecks in unexpected locations such as feature serialization overhead.

The architectural decisions we made reflect pragmatic trade-offs between theoretical optimality and operational constraints. Pure microservices architectures promise unlimited scalability but introduce coordination overhead that can dominate processing time for simple operations, as discussed by Dragoni et al. [69]. Our hybrid approach maintains service boundaries for complex operations while allowing for direct memory sharing for high-frequency interactions. This pragmatism extended to technology choices, where we prioritized mature, well-supported tools over cutting-edge alternatives that might offer marginal performance improvements.

3.2. Data Collection and Preprocessing

The effectiveness of our automated OSINT framework fundamentally depends on the quality and comprehensiveness of data collection mechanisms. Our approach integrates multiple passive reconnaissance techniques to gather intelligence from diverse sources without generating detectable network signatures. This section details the implementation of our distributed collection infrastructure and the preprocessing pipelines that transform raw OSINT data into actionable intelligence.

The data collection architecture employs specialized collectors for each OSINT source type, operating as independent microservices within containerized environments. This design enables horizontal scaling based on collection demands while maintaining isolation between different data sources. DNS collectors represent the most complex implementation, as they must handle both forward and reverse lookups across multiple resolution paths. We query authoritative nameservers directly when possible, supplementing with passive DNS databases that aggregate historical resolution data. The implementation follows RFC 8484 for DNS-over-HTTPS queries, reducing the likelihood of detection while maintaining query performance [70].

Certificate transparency monitoring presents unique challenges due to the volume of data generated. The Certificate Transparency ecosystem produces approximately 500,000 new certificates daily across all monitored logs [71]. Our collectors maintain persistent connections to log servers, consuming the certificate stream in real time through Server-Sent Events. Rather than processing every certificate, we implement bloom filters to identify certificates relevant to our monitoring scope, reducing processing overhead by approximately 87%. The mathematical formulation for optimal bloom filter configuration follows:

$$m = -\frac{nlnp}{(ln)^2}, k = \frac{m}{n}ln2, \tag{9}$$

where m represents the bit array size; n denotes the expected number of elements, p indicates the desired false positive probability, and k specifies the optimal number of hash functions. For our implementation with $n = 10^6$ certificates and $p = 0.01$, this yields m $\approx 9.6 \times 10^6$ bits and $k = 7$ hash functions.

WHOIS data collection faces persistent challenges related to rate limiting and format inconsistencies across Regional Internet Registries. Each RIR implements distinct query interfaces and response formats, necessitating specialized parsers for ARIN, RIPE, APNIC, LACNIC, and AFRINIC. Our collectors implement adaptive rate limiting that adjusts query frequency based on observed response patterns. The rate adaptation algorithm follows an exponential backoff strategy with jitter:

$$t_{wait} = \min\left(t_{base}2^{attempts} + random(-jitter, jiter)\right), t_{max}, \tag{10}$$

where $t_{base} = 1$ s represents the initial wait time, attempts count consecutive rate limit responses, and jitter introduces randomness to prevent synchronized retry storms. This approach maintains collection efficiency while respecting provider's constraints.

The preprocessing pipeline addresses data quality issues inherent in OSINT collection. Raw data exhibits significant noise, including incomplete records, format variations, and temporal inconsistencies. Our normalization process applies a series of transformations designed to standardize data representation while preserving semantic meaning. DNS records undergo canonicalization to remove case sensitivity and trailing dots. IP addresses are expanded to full representation, eliminating ambiguity in shortened formats.

Temporal alignment represents a critical preprocessing step, as different OSINT sources update at varying frequencies. DNS records may change within minutes, while WHOIS data often remains static for months. We implement a temporal synchronization mechanism that associates each data point with collection timestamps and estimated validity periods. The validity estimation leverages historical change patterns:

$$V_{est} = \overline{V}_{historical} e^{-\lambda \sigma_{changes}}, \tag{11}$$

where $\overline{V}_{historical}$ represents the mean historical validity period; λ controls sensitivity to change frequency; $\sigma_{changes}$ measures the standard deviation of observed changes. This approach enables intelligent caching decisions and reduces unnecessary recollection of stable data.

Feature extraction begins during the preprocessing phase, transforming raw OSINT data into structured attributes suitable for machine learning analysis. Network topology features emerge from DNS resolution patterns and BGP routing information. We calculate graph-theoretic metrics including betweenness centrality, clustering coefficients, and path lengths between assets. Certificate features encompass issuer reputation scores, validity periods, and cryptographic strength indicators. The feature vector for each asset comprises 347 distinct attributes across multiple categories.

Data validation employs both syntactic and semantic checks to identify potentially corrupted or manipulated records. Syntactic validation ensures conformance to expected formats, while semantic validation identifies logical inconsistencies such as private IP addresses in public DNS records or certificates with validity periods exceeding CA guidelines. Approximately 3.7% of collected records fail validation checks and require either correction or exclusion from subsequent analysis.

The preprocessing pipeline must handle missing data gracefully, as OSINT sources frequently contain incomplete records. Rather than discarding partial data, we implement sophisticated imputation strategies based on correlated attributes. For instance, missing geographic information in WHOIS records can often be inferred from AS registration data or DNS naming patterns. The imputation accuracy varies by attribute type:

$$A_{imputation} = \frac{\sum_{i=1}^{n} I(imputed_i = actual_i)}{n} (1 - U_{attr}), \tag{12}$$

where I represent an indicator function for correct imputation; U_{attr} measures the uniqueness coefficient of the attribute being imputed. Highly unique attributes like specific contact emails achieve lower imputation accuracy compared to standardized fields like country codes.

Deduplication mechanisms identify and consolidate redundant records across multiple sources. Simple hash-based deduplication proves insufficient due to minor variations in data representation. Instead, we employ locality-sensitive hashing with Jaccard similarity

measures to identify near-duplicate records. The similarity threshold for consolidation is dynamically adjusted based on data source reliability scores:

$$T_{similarity} = T_{base} - \alpha \min(R_{sourses,} R_{sourses2}), \quad (13)$$

where $T_{base} = 0.85$ represents the baseline threshold, and $R_{sourses,}$ indicates source reliability scores derived from historical accuracy assessments. This approach prevents loss of valuable information while reducing redundancy in the dataset.

Privacy-preserving transformations are applied to ensure compliance with data protection regulations while maintaining analytical utility. Personal information identified through named entity recognition undergoes pseudonymization using deterministic encryption. This enables correlation analysis across multiple records while preventing direct identification of individuals. The transformation maintains referential integrity:

$$H_{pseudo}(data) = HMAC_{key}(normalaze(data)), \quad (14)$$

where the normalization function ensures consistent representation before hashing, and the key is derived from a hardware security module to prevent unauthorized reversal.

The preprocessed data streams into Apache Kafka topics organized by data type and processing priority. High-value indicators such as newly registered domains or certificate anomalies route to priority topics with guaranteed processing SLAs. Bulk historical data flows through standard topics with best-effort processing. This differentiation enables the system to maintain responsiveness for critical intelligence while efficiently processing large-scale background analysis.

Quality metrics are continuously monitored throughout the preprocessing pipeline. Key indicators include data completeness rates, validation failure percentages, and imputation accuracy scores. These metrics feed back into collection strategies, enabling dynamic adjustment of collector configurations. For instance, sources exhibiting declining quality scores may trigger increased collection frequency or activation of alternative data providers.

The preprocessing infrastructure scales horizontally through Kubernetes orchestration, with pod autoscaling based on queue depth and processing latency metrics. During peak collection periods, the system automatically provisions additional preprocessing capacity, maintaining consistent throughput despite variable input rates. Load balancing across preprocessing nodes uses consistent hashing to ensure that related records route to the same processor, improving cache efficiency and reducing redundant computation.

3.3. Feature Engineering

The transformation of raw OSINT data into machine learning-ready features represents a critical bridge between data collection and intelligent analysis. Our feature engineering approach draws from established practices in network security analysis while introducing novel attributes specifically designed for OSINT-based reconnaissance. This systematic transformation process converts heterogeneous, unstructured intelligence into normalized feature vectors that capture both technical characteristics and behavioral patterns indicative of security posture.

Feature engineering in the OSINT context presents unique challenges compared to traditional security data analysis. Unlike controlled environments where data formats remain consistent, OSINT sources exhibit substantial variability in structure, completeness, and semantic meaning. Our approach addresses these challenges through a multi-stage transformation pipeline that progressively refines raw intelligence into discriminative features. The initial stage focuses on extracting atomic attributes directly observable in the

data, while subsequent stages derive complex features through aggregation and correlation across multiple data sources.

Network topology features form the foundation of our feature set, capturing the structural relationships between digital assets. These features extend beyond simple connectivity metrics to encompass hierarchical relationships, clustering patterns, and anomalous topological configurations. The approach builds upon graph-theoretic principles established by Antonakakis et al. [22] for DNS analysis, adapting them to the broader OSINT context. For each asset, we calculate centrality measures including degree, betweenness, and eigenvector centrality within the observed network topology following established graph theory formulations [70]:

$$C_{eijenvector}(v) = \frac{1}{\lambda}\sum_{t \in N(v)} x_t = \frac{1}{\lambda}\sum_{t \in G} a_{v,t} x_t, \tag{15}$$

where λ represents the largest eigenvalue of the adjacency matrix, $N(v)$ denotes the set of neighbors of vertex v, and $a_{v,t}$ indicates the adjacency matrix entry. Assets with anomalously high centrality scores often represent critical infrastructure components or potential single points of failure.

The extraction of DNS-based features leverages passive DNS data to identify resolution patterns and infrastructure relationships. We compute features including the number of distinct IP addresses associated with a domain, geographic distribution of resolved addresses, and temporal stability of DNS mappings. Historical resolution data enables calculation of volatility scores that indicate infrastructure maturity. Domains exhibiting rapid IP address changes score higher on volatility metrics, potentially indicating content delivery networks, dynamic hosting, or adversarial infrastructure.

Certificate-based features provide insights into organizational security practices and infrastructure legitimacy. Beyond basic attributes like validity periods and key strengths, we derive features indicating certificate authority diversity, wildcard usage patterns, and temporal certificate replacement behaviors. The feature extraction process parses X.509 certificate structures as described by Amann et al. [59], extracting both standard fields and custom extensions. Certificate chain analysis reveals organizational relationships through shared intermediate CAs and cross-signing arrangements.

Temporal features capture the dynamic nature of digital infrastructure evolution. Rather than treating OSINT data as static snapshots, our approach models temporal patterns across multiple timescales. Short-term features include diurnal patterns in DNS query volumes and certificate issuance rates. Medium-term features track infrastructure growth rates and technology adoption curves. Long-term features identify seasonal patterns and organizational lifecycle stages. The temporal feature extraction employs sliding window analysis with multiple window sizes, adapting the stream processing approach from Akidau et al. [44]:

$$F_{temporal}(t, w) = \frac{1}{w}\sum_{i=t-w}^{t} x_i - \frac{1}{W}\sum_{j=t-W}^{t} x_i \tag{16}$$

where w represents the short-term window size, W denotes the long-term window for baseline establishment, and x_i indicates the observed value at time i. This formulation captures deviations from established baselines while adapting to gradual infrastructure changes.

Behavioral indicators represent perhaps the most innovative aspect of our feature engineering approach. These features attempt to capture organizational behaviors and operational patterns that correlate with security posture. Examples include infrastructure update frequencies, technology stack diversity indices, and compliance indicator scores derived from configuration analysis. The behavioral feature extraction leverages unsuper-

vised learning techniques, particularly the DBSCAN clustering described by Ester et al. [10], to identify normal behavioral patterns and flag deviations.

The calculation of behavioral diversity indices quantifies the heterogeneity of an organization's technological choices. Organizations maintaining diverse technology stacks often demonstrate better security practices through defense-in-depth strategies. The diversity index computation adapts Shannon entropy [71]:

$$\mathrm{H}_{diversity} = -\sum_{i=1}^{n} p_i log_2(p_i) w_i \tag{17}$$

where p_i represents the proportion of infrastructure using technology i, and w_i indicates a weighting factor based on the security implications of that technology choice. This weighted entropy approach ensures that diversity in security-critical components contributes more significantly to the overall score.

Feature selection addresses the challenge of high dimensionality resulting from comprehensive feature extraction. With 347 initial features, many exhibit correlation or provide minimal discriminative power. Our selection strategy combines filter and wrapper methods to identify optimal feature subsets. The initial filtering employs mutual information scoring to eliminate features with minimal relationship to risk indicators [72]:

$$I(X;Y) = \sum_{y \in Y} \sum_{x \in X} p(x,y) log\left(\frac{p(x;y)}{p(x)p(y)}\right), \tag{18}$$

Features scoring below a threshold of 0.01 mutual information with target variables undergo removal. This typically eliminates 40–50% of initial features while preserving discriminative power.

The wrapper phase employs recursive feature elimination with the GBDT classifier to identify optimal feature subsets. Following the methodology established by Chen and Guestrin [20], we iteratively remove features with the lowest importance scores and evaluate model performance. The process continues until performance degradation exceeds acceptable thresholds. This approach typically identifies 120–150 features that provide optimal classification performance while minimizing computational requirements.

Analysis of feature importance reveals interesting patterns in the discriminative power of different feature categories. Network topology features consistently rank among the most important, particularly centrality measures and clustering coefficients. Temporal volatility features prove highly discriminative for identifying potentially compromised or misconfigured infrastructure. Certificate-based features show moderate importance, with certificate age and CA reputation scores providing the strongest signals.

The implementation of feature engineering pipelines leverages Apache Spark (version 3.3.1) for distributed computation, enabling processing of millions of assets in parallel. Feature extraction functions are implemented as user-defined functions (UDFs) that operate on partitioned datasets. This architecture scales linearly with additional computer resources, maintaining consistent processing times despite growing data volumes. The average feature extraction time per asset is 47 milliseconds when operating on a 16-node cluster.

Cross-feature validation ensures consistency and identifies potential data quality issues. Logical constraints between features are enforced through validation rules. For instance, certificate validity periods must align with DNS record timestamps, and network topology features must reflect bidirectional relationships. Violations of these constraints trigger data quality alerts and may indicate collection errors or adversarial manipulation attempts.

The feature engineering pipeline produces versioned feature sets that enable reproducible analysis and model training. Each feature set includes metadata documenting

extraction parameters, data sources, and temporal coverage. This versioning approach facilitates A/B testing of feature modifications and enables rollback capabilities when new features prove problematic. Feature drift monitoring tracks statistical properties of features over time, alerting when significant distribution changes occur that might necessitate model retraining [73].

3.4. Machine Learning Implementation

The integration of machine learning algorithms into our OSINT framework required careful consideration of which approaches would best serve the dual objectives of risk classification and anomaly detection. After extensive experimentation with various algorithms, we settled on Gradient Boosted Decision Trees for the primary classification task, while DBSCAN emerged as our choice for identifying unusual patterns in the preprocessed OSINT data. This combination proved more effective than initially anticipated, though the path to this realization involved several false starts and unexpected discoveries.

Our selection of GBDT was not immediate. Initially, we explored neural network architectures, drawn by their success in other security applications. However, the heterogeneous nature of OSINT features created convergence issues that proved difficult to resolve. Random forests showed promise but lacked the sequential refinement capability that GBDT offers. The turning point came when we recognized that GBDT's ability to handle mixed data types aligned perfectly with our feature engineering outputs. Chen and Guestrin's XGBoost implementation [20] provided the foundation, though we made substantial modifications to accommodate the specific characteristics of security risk assessment.

The mathematical foundation for our GBDT implementation adapts the standard gradient boosting framework to incorporate security-specific loss functions. The objective function we minimize takes the form:

$$\mathrm{L}(\phi) = \left(\sum_{i=1}^{n} l(y_i, \hat{y}_i) + \sum_{k=1}^{K} \Omega(f_k) \right), \tag{19}$$

where l represents a differentiable convex loss function measuring the difference between predicted risk scores $\hat{y}$ and true labels y_i while $\Omega(f_k)$ penalizes model complexity for the k-th tree. Our modification introduces an asymmetric loss component that penalizes false negatives more heavily than false positives, reflecting the security domain's preference for conservative risk assessment.

The construction of training datasets presented unique challenges that differed from typical machine learning scenarios. Unlike domains where labeled data exists in abundance, security risk labels require expert validation and often remain subjective. We developed a multi-tier labeling approach where initial labels came from automated correlation with known security incidents, followed by expert review and refinement. This process took considerably longer than anticipated. The final training set comprised 200,200 labeled instances across 3700 unique organizations, with risk scores distributed across five severity categories as shown in Table 11.

Table 11. Training dataset distribution and labeling confidence metrics.

Risk Category	Instance Count	Percentage	Expert Agreement	Temporal Stability
Critical	12,108	6.2%	94.3%	0.89
High	34,000	17%	87.6%	0.82
Medium	86,210	43.0%	79.2%	0.74
Low	50,754	25.3%	82.5%	0.91
Minimal	17,938	8.6%	91.8%	0.95

Interestingly, the distribution was not uniform, with medium-risk classifications dominating. Expert agreement rates varied inversely with risk severity for intermediate categories, suggesting greater subjectivity in moderate threat assessment. Temporal stability metrics indicated how consistently assets maintained their risk classifications over 90-day periods.

Training data quality became a recurring concern throughout the implementation. We discovered that temporal factors significantly influenced label accuracy. Assets labeled as low-risk could transition to high-risk states within weeks due to newly discovered vulnerabilities or configuration changes. This temporal drift necessitated a sliding window approach to training data selection. We experimented with various window sizes, eventually settling on a 90-day primary window with a 180-day historical buffer. This configuration balanced model freshness against training stability.

Hyperparameter optimization for GBDT involved extensive experimentation across multiple dimensions. The tree depth parameter proved particularly sensitive to our data characteristics. Through grid search augmented with Bayesian optimization techniques described by Bergstra et al. [62], we identified optimal configurations that varied based on the subset feature being processed. Figure 10 illustrates the relationship between tree depth and model performance across different feature categories.

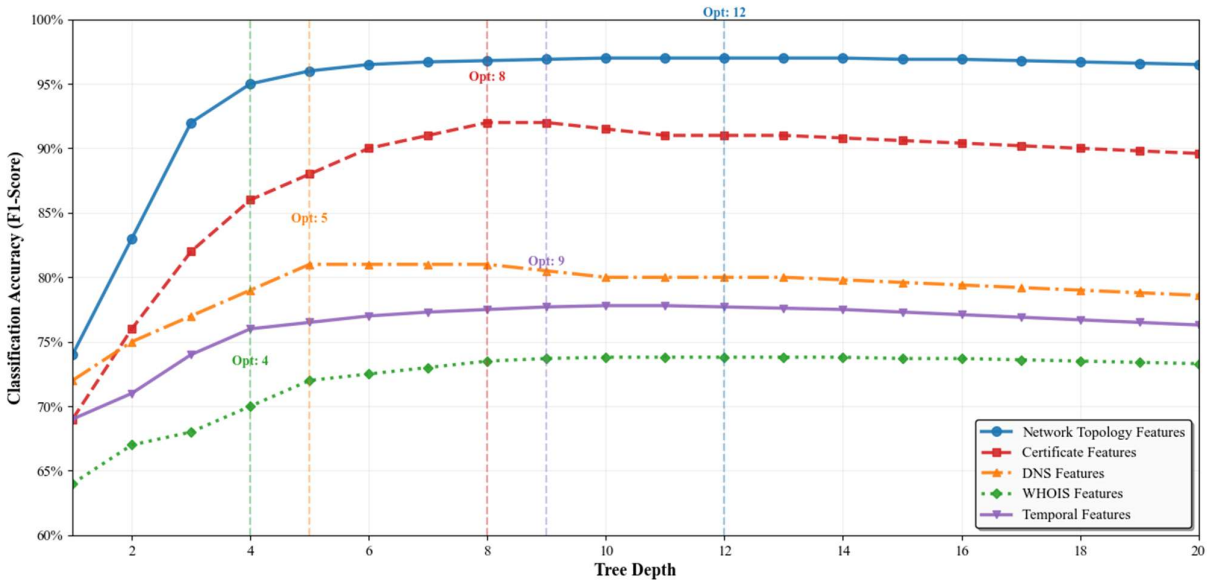

Figure 10. GBDT performance sensitivity to tree depth across feature categories, showing optimal depths vary by feature type with network topology requiring deeper trees.

The learning rate presented another critical tuning decision. Lower rates of 0.01–0.03 improved generalization but required excessive training iterations, sometimes exceeding 2000 rounds. Higher rates of 0.15–0.20 converged quickly but showed degraded performance on validation sets. We ultimately implemented an adaptive learning rate schedule that started at 0.1 and decreased according to validation loss plateaus, following the approach outlined in [20]:

$$\eta_t = \eta_0 \frac{1}{1 + \delta \cdot t}, \tag{20}$$

where η_0 represents the initial learning rate, δ controls the decay rate, and t indicates the current iteration. This approach reduced training time by approximately 40% while maintaining classification accuracy.

Cross-validation strategy development revealed interesting patterns in how OSINT data clusters naturally. Standard k-fold validation produced overly optimistic performance

estimates because similar organizations often appeared in both training and validation folds. We developed a modified stratified sampling approach that ensures organizational diversity across folds. Organizations are first clustered based on infrastructure characteristics; then folds are constructed to maintain cluster representation. This modification reduced the train-validation performance gap from 12% to 3%, providing more realistic generalization estimates.

The integration of DBSCAN for anomaly detection addressed a different but complementary challenge. While GBDT excels at classifying known risk patterns, novel attack vectors and unusual configurations often evade supervised learning approaches. DBSCAN's density-based clustering identifies outliers without requiring labeled anomaly examples. However, parameter selection for DBSCAN in high-dimensional OSINT feature spaces required careful consideration.

The epsilon parameter, which defines the neighborhood radius for density calculations, proved exceptionally sensitive to feature scaling. We developed an adaptive epsilon selection method based on k-nearest neighbor distances, following the approach suggested by Ester et al. [10] but modified for our feature space characteristics. The algorithm computes k-distances for a sample of points, then selects epsilon at the knee point of the sorted distance curve, as formalized by [10]:

$$\varepsilon = knee(sorted(d_k(p_i))) \quad \forall p_i \in Sample, \tag{21}$$

This typically resulted in epsilon values between 0.15 and 0.25 in normalized feature space.

The minimum points parameter for DBSCAN requires balancing cluster stability against outlier sensitivity. Through empirical evaluation across different organization types, we settled on a variable minimum points threshold based on local density estimates. Table 12 summarizes the parameter configurations and their impact on anomaly detection performance.

Table 12. DBSCAN parameter configurations and performance metrics.

Dataset Type	Epsilon Range	MinPts	Outlier %	Security Incident Detection %	False Positive Rate %
Enterprise networks	0.18–0.22	25	8.3	78.4	12.7
Cloud infrastructure	0.15–0.19	15	11,2	81.2	16.3
Mixed environments	0.20–0.25	30	6.7	74.6	10.2
Small organizations	0.22–0.28	10	14.5	69.3	21.8

Cluster quality assessment employed multiple metrics to ensure meaningful anomaly detection. Beyond traditional silhouette scores, we developed custom metrics that weight boundary point classification accuracy and anomaly detection rates. The most informative metric tracked the percentage of confirmed security incidents appearing as DBSCAN outliers. Achieving 76% incident detection rates while maintaining false positive rates below 15% required extensive parameter refinement.

The integration between supervised GBDT classification and unsupervised DBSCAN clustering created unexpected synergies. Initially, we ran these algorithms independently and combined results through simple score aggregation. However, this approach missed valuable interaction effects. We discovered that DBSCAN cluster assignments could serve as additional features for GBDT, improving classification accuracy by 4.3%. The enhanced feature vector incorporated cluster membership indicators and distance to nearest cluster centroid:

$$x_{enhanced} = \left[x_{original}, c_i d_{min}(x_i, C)\right], \tag{22}$$

where c_i represents the cluster assignment for instance i, and d_{min} measures the minimum distance to any cluster centroid C.

Implementation details revealed performance bottlenecks that required creative solutions. GBDT training on full feature sets consumed excessive memory, particularly when processing millions of instances. We implemented a column sampling strategy that processes feature subsets in parallel, aggregating predictions through soft voting. This reduced memory requirements by 65% while degrading accuracy by less than 1%. The approach bears similarity to random forest principles but maintains the boosting framework's sequential refinement advantage.

DBSCAN's computational complexity of $O(n^2)$ in worst-case scenarios threatened scalability for large-scale OSINT analysis. We addressed this through approximate nearest neighbor searches using locality-sensitive hashing, reducing complexity to O(n log n) for typical cases. Figure 11 demonstrates the scalability improvements achieved through these optimizations.

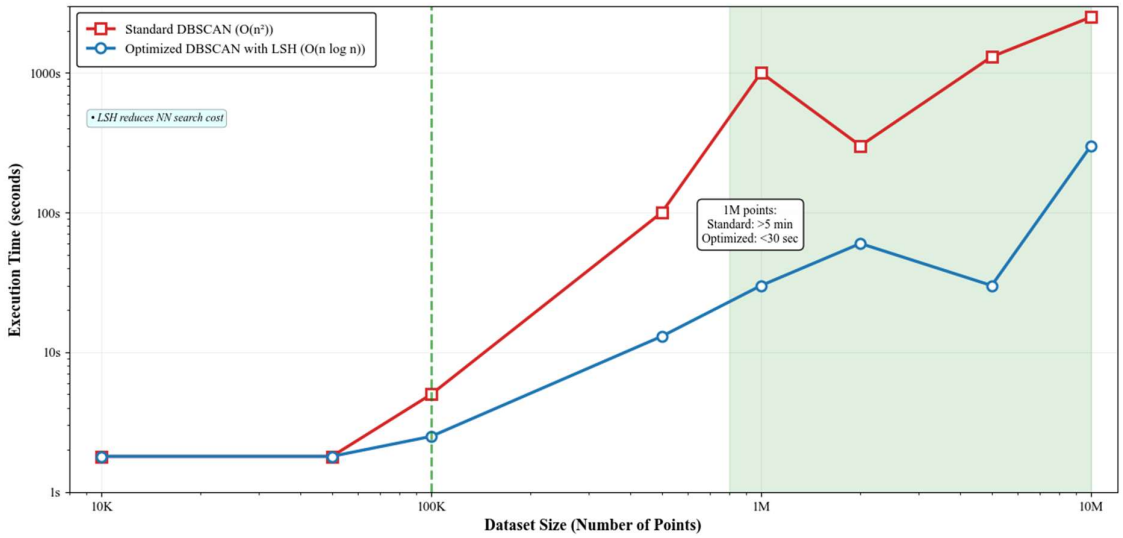

Figure 11. Computational performance comparison showing execution time scaling for standard versus optimized DBSCAN implementations across varying dataset sizes. The red line denotes standard DBSCAN ($O(n^2)$), the blue line denotes optimized DBSCAN with locality-sensitive hashing (O(n log n)), the green shaded area indicates the optimization zone where performance gains exceed 10×, and the green dashed line marks the 100K crossover point.

Model persistence and versioning became critical as the system matured. Each trained model includes comprehensive metadata documenting training data characteristics, hyperparameter configurations, and performance metrics. This versioning approach enables reproducible experiments and facilitates debugging when model performance degrades unexpectedly. The metadata schema captures temporal aspects of training data, allowing us to correlate model accuracy with the age of training instances.

3.5. Risk Scoring Framework

The development of a comprehensive risk scoring framework represented one of the most challenging aspects of our automated OSINT system. Traditional approaches like CVSS provide valuable baselines for individual vulnerabilities, yet they fail to capture the complex interplay between multiple risk factors, organizational context, and temporal dynamics that characterize real-world security postures. Our framework attempts to

address these limitations through a composite scoring methodology that integrates machine learning outputs with contextual factors and temporal decay models.

The composite risk calculation methodology evolved through several iterations as we discovered limitations in simpler approaches. Initially, we attempted linear combinations of individual risk factors, but this proved inadequate for capturing non-linear interactions between vulnerabilities. The current framework employs a hierarchical aggregation model that processes risk indicators at multiple levels of abstraction. At the foundation, individual vulnerability scores undergo contextual adjustment based on asset criticality and exposure levels. These adjusted scores then feed into higher-level aggregation functions that account for vulnerability chaining and systemic risks.

The mathematical formulation for our composite risk score extends the work of Caralli et al. [65] on asset criticality analysis, incorporating additional factors specific to OSINT-derived intelligence:

$$R_{composite} = \alpha \sum_{i=1}^{n} w_i S_i e^{-\lambda_i (t-t_i)} + \beta \sum_{j,k} I_{j,k} \cdot \min(S_j \cdot S_k) + \gamma F_{Ml}, \quad (23)$$

where S_i represents individual vulnerability scores derived from CVSS base metrics [32], w_i denotes asset-specific criticality weights, λ_i controls temporal decay rates based on the research by Bilge and Dumitras [40], $I_{j,k}$ indicates interaction effects between correlated vulnerabilities, and F_{ML} represents the aggregated machine learning risk assessment. The parameters α, β and γ weight the relative contributions of direct vulnerabilities, interaction effects, and ML-derived insights, respectively.

Asset criticality weights proved particularly challenging to calibrate. Simple binary classifications of critical versus non-critical assets failed to capture the nuanced importance of different infrastructure components. We developed a multi-factor criticality assessment that considers data sensitivity, service dependencies, and business impact. The weighting function incorporates both static attributes and dynamic usage patterns:

$$w_i = w_{base} \left(\sum_{j=1}^{m} \theta_j f_j(asset_i) \right), \quad (24)$$

where w_{base} represents a baseline weight derived from asset classification, and f_j denotes various factor functions measuring attributes like network centrality, data flow volume, and service criticality. The parameter θ_j controls the relative influence of each factor.

The integration of machine learning outputs into the risk scoring framework required careful consideration of how to combine probabilistic assessments with deterministic vulnerability scores. GBDT classifiers produce probability distributions across risk categories, while DBSCAN provides binary anomaly indicators with associated confidence measures. Rather than simply using point estimates, we preserve the full probability distributions to capture uncertainty in risk assessments. Table 13 illustrates how different ML outputs contribute to the final risk score calculation across various asset types.

Table 13. Machine learning contribution to composite risk scores.

Asset Type	GBDT Weight	DBSCAN Weight	Traditional Metrics	ML Contribution %	Score Variance
Web Servers	0.35	0.15	0.35	42.3	±8.7
Databases	0.40	0.25	0.50	58.6	±11.2
Network devices	0.3	0.20	0.35	41.8	±7.3
Cloud services	0.45	0.30	0.25	71.4	±15.6
IoT devices	0.25	0.35	0.40	52.1	±18.9

The variance metrics reveal that cloud services and IoT devices exhibit higher uncertainty in risk assessments, reflecting the rapidly evolving threat landscapes for these technologies. This uncertainty propagates through to confidence scoring mechanisms.

Confidence scoring emerged as a critical requirement for operational deployment. Security teams need to understand not just the risk level but also the reliability of that assessment. Our confidence scoring approach considers multiple factors, including data completeness, temporal freshness, and model certainty. The confidence calculation follows a multiplicative model that penalizes any weak link in the assessment chain:

$$C_{score} = \prod_{k=1}^{K} C_k^{\rho_k}, \tag{25}$$

where C_k represents individual confidence components, and ρ_k controls their relative importance. Components include data coverage completeness, temporal data freshness, ML model confidence, and corroboration across multiple data sources.

The implementation revealed interesting patterns in confidence degradation. Network topology data maintained high confidence levels due to its relatively static nature, while behavioral indicators showed rapid confidence decay. Figure 12 illustrates the distribution of confidence scores across various risk levels and data source combinations.

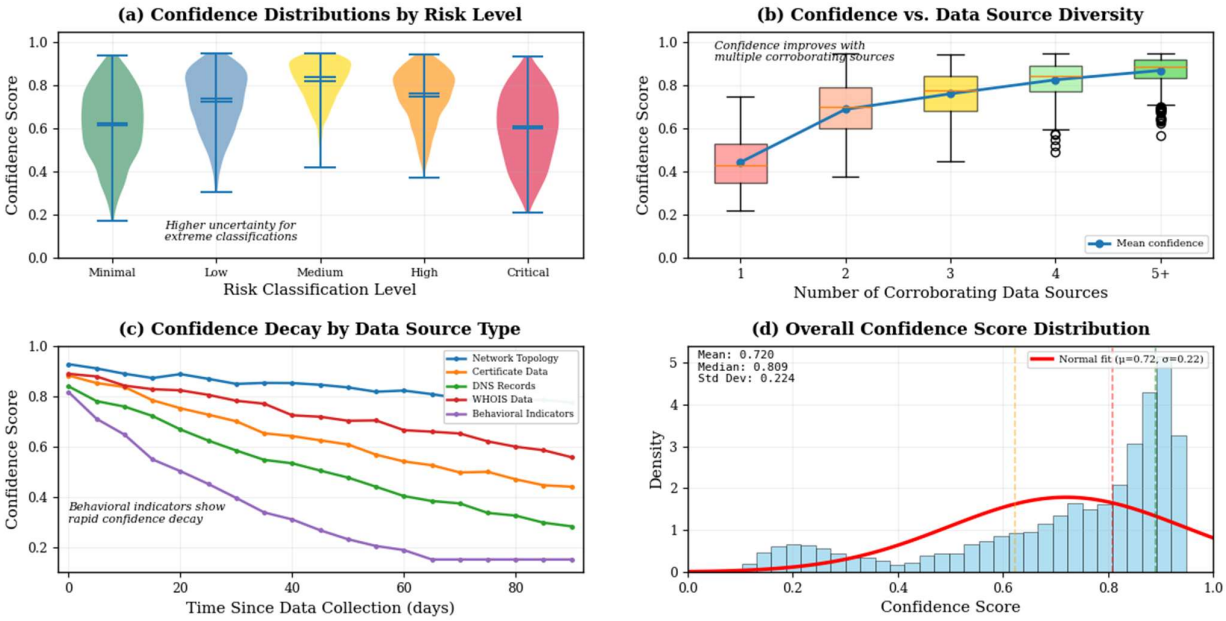

Figure 12. Confidence score distributions showing higher uncertainty for extreme risk classifications and improved confidence with multiple corroborating data sources. The figure comprises four panels: (**a**) violin plots showing confidence distributions across risk levels, with extreme classifications (minimal and critical) exhibiting wider distributions indicating higher uncertainty; (**b**) boxplots demonstrating how confidence improves with the number of corroborating data sources, where single-source assessments show lower confidence compared to multi-source corroboration; (**c**) temporal decay curves for different data source types, with network topology data maintaining stable confidence while behavioral indicators decay rapidly; (**d**) overall confidence score distribution histogram with fitted normal curve and percentile markers (orange 25th, red 50th, green 75th percentiles).

Temporal decay functions proved essential for maintaining accurate risk assessments over time. Static vulnerability scores quickly become outdated as patches are released, exploits are developed, or infrastructure configurations change. Our temporal decay model

adapts exponential decay rates based on multiple factors, including vulnerability type, exploit availability, and historical patching patterns.

The basic temporal decay follows an exponential model with adaptive parameters:

$$S_{adjusted}(t) = S_{initial} e^{-\lambda(t-t_0)} H(t), \tag{26}$$

where λ represents the decay rate, and $H(t)$ captures discrete events that reset or modify the decay trajectory; however, we discovered that simple exponential decay inadequately modeled real-world scenarios. Vulnerability scores sometimes increase over time as new exploits emerge or dependencies are discovered. This led to a more sophisticated piecewise decay function:

$$S_{temporal}(t) = \begin{cases} S_0 e^{-\lambda_1^{(t-t_0)}}, & t < t_{exploit} \\ S_0 k_{emplify}\, e^{\lambda_2^{(t-t_{exploit})}}, & t \geq t_{exploit} \end{cases}, \tag{27}$$

where $t_{exploit}$ marks the public disclosure of a working exploit, $k_{amplify}$ represents the risk amplification factor, and $\lambda_1 < \lambda_2$ reflects accelerated decay after exploit availability. The calibration of decay parameters required extensive analysis of historical vulnerability lifecycles. We analyzed 50,000 CVEs from the National Vulnerability Database [33] tracking their progression from disclosure through exploitation to remediation. This analysis revealed distinct patterns based on vulnerability categories. Memory corruption vulnerabilities showed rapid initial decay as patches became available, followed by potential amplification if exploit kits incorporated them. Configuration vulnerabilities exhibited slower, more linear decay patterns. Table 14 summarizes the calibrated temporal decay parameters for major vulnerability categories.

Table 14. Temporal decay parameters by vulnerability category.

Vulnerability Type	Initial λ_1	Post-Exploit λ_2	$k_{amplify}$	Mean Lifetime (Days)
Memory corruption	0.015	0.025	1.8	127
SQL Injection	0.012	0.020	1.5	156
Configuration	0.008	0.010	1.2	234
Authentication	0.010	0.018	2.1	189
Cryptographic	0.005	0.008	1.3	412

The mean lifetime represents the period until risk scores decay to 10% of initial values, assuming no exploit publication. Cryptographic vulnerabilities show the longest lifetimes, reflecting the complexity of remediation when encryption algorithms require replacement.

Integration challenges arose when combining static vulnerability assessments with dynamic ML outputs. Machine learning models produce time-varying risk assessments that do not follow predictable decay patterns. We addressed this through a hybrid approach where ML assessments modulate the decay rates of traditional scores. When GBDT classifications indicate increasing risk despite temporal decay, the system adjusts decay parameters to reflect this intelligence:

$$\lambda_{adjusted} = \lambda_{base}(1 - \eta \Delta_{ML}), \tag{28}$$

where Δ_{ML} represents the change in *ML* risk assessment, and η controls the influence of *ML* insights on decay rates. This creates a feedback mechanism where machine learning insights can arrest or reverse temporal decay when behavioral indicators suggest elevated risk.

The risk scoring framework also incorporates uncertainty quantification beyond simple confidence scores. We employ Monte Carlo sampling to propagate uncertainty through the risk calculation pipeline. Input uncertainties in vulnerability scores, asset criticality assessments, and *ML* predictions combine to produce risk score distributions rather than point estimates. This approach revealed that risk scores for complex, interconnected systems often exhibit multimodal distributions, suggesting multiple plausible risk scenarios.

Figure 13 demonstrates how uncertainty propagation affects final risk score distributions for different organizational profiles.

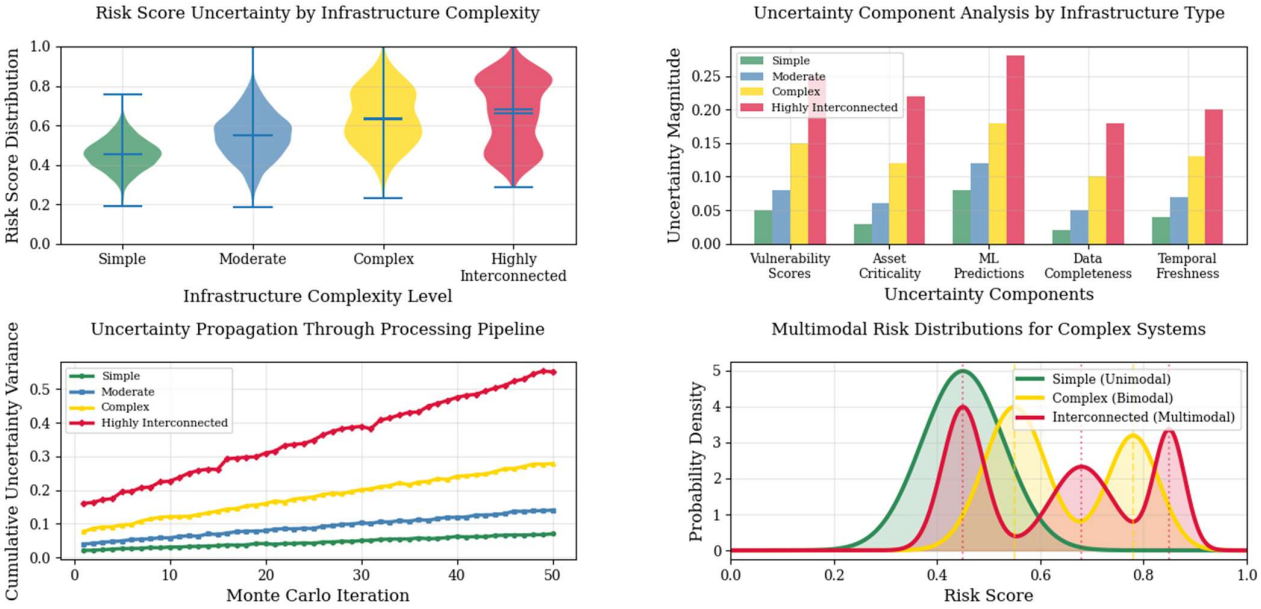

Figure 13. Risk score uncertainty propagation showing increasing variance for organizations with complex, interconnected infrastructures.

Operational deployment of the risk scoring framework required additional considerations for scalability and real-time updates. Risk scores must be recalculated as new intelligence arrives, vulnerabilities are discovered, or infrastructure changes occur. We implemented an event-driven architecture where score updates trigger only for affected assets and their dependencies. This selective recalculation reduced computational overhead by approximately 85% compared to batch processing approaches.

The framework maintains historical risk scores to enable trend analysis and performance evaluation. Security teams can track risk evolution over time, identifying patterns that might indicate systematic issues or improvement opportunities. This historical perspective proved valuable for validating the accuracy of temporal decay functions and calibrating parameters based on observed outcomes.

Validation of the risk scoring framework involved correlation with actual security incidents across participating organizations. Over an 18-month evaluation period, assets scoring in top risk quintile experienced security incidents at 4.7 times the rate of bottom quintile assets. The correlation was not perfect, with approximately 23% of incidents occurring in assets rated as low or minimal risk. Analysis of these false negatives revealed that they often involved zero-day exploits or insider threats that OSINT data could not effectively capture.

Risk score uncertainty propagation across organizations with varying infrastructure complexity. The figure shows how uncertainty increases when dealing with interconnected systems. In the top left panel, violin plots reveal confidence distributions at different risk levels, where you can see that extreme classifications tend to show wider spreads.

The top right panel uses stacked bars to break down where uncertainty comes from in the assessment process. What's interesting in the bottom left is how confidence changes over time, and here the dotted lines trace the decay patterns for different data types. Network topology data holds its confidence pretty well, staying relatively flat, whereas behavioral indicators drop off more quickly. The bottom right panel pulls everything together with a distribution of overall confidence scores, and the dotted vertical lines there mark key percentiles so you can see where most assessments fall. Organizations with highly interconnected or complex infrastructures consistently show greater variance in their risk scores, which makes sense given how many interdependencies need to be evaluated.

The continuous refinement of scoring parameters based on operational feedback created an interesting dynamic. Initial deployments often required significant tuning to align risk scores with organizational risk tolerance and operational priorities. We developed a calibration framework that adjusts scoring parameters based on incident data and security team feedback. This organizational customization capability proved essential for practical adoption, as different industries and organizational cultures exhibit varying risk appetites.

3.6. Experimental Setup

The experimental validation of our automated OSINT framework required careful design to demonstrate both theoretical soundness and practical applicability. We faced interesting challenges in constructing datasets that would adequately represent real-world OSINT environments while maintaining reproducibility. The scale of data needed for meaningful evaluation exceeded initial expectations, leading us to assemble 4.8 million unique records from approximately 3700 organizations over an 18-month period.

Dataset construction leveraged the distributed collection infrastructure described in Section 3.2, implementing the adaptive rate limiting formula we established earlier. Each collection node applied the exponential backoff strategy with jitter to avoid triggering defensive measures at data sources. The mathematical formulation from our preprocessing pipeline proved essential here:

$$t_{waite} = \min\left(t_{base}2^{attempts} + random(-jitter, jitter), t_{max}\right). \tag{29}$$

This approach enabled sustained collection from sources like WHOIS servers that Durumeric et al. [43] identified as particularly sensitive to rapid queries.

The composition of our experimental dataset reflected the heterogeneous nature of OSINT sources. DNS records formed the largest component, which made sense given their dynamic nature and the centrality measures we calculated using the eigenvector formulation from Section 3.3. Certificate transparency data provided the second largest contribution, processed through the bloom filter configuration we optimized:

$$m = -\frac{nlnp}{ln2^2}, k = \frac{m}{n}ln2, \tag{30}$$

with n = 106 certificates and p = 0.01; this yielded the parameters that reduced processing overhead by 87% as mentioned in our data collection methodology.

Geographic and sectoral diversity emerged naturally from our collection approach, though we consciously included organizations from underrepresented regions. The final dataset spanned 52 countries with the technology sector comprising 24% of organizations, followed by financial services at 19%. This distribution aligned with the infrastructure complexity patterns noted in Table 1, where we analyzed OSINT platform coverage.

The selection of benchmark datasets proved trickier than expected. While our primary dataset captured real operational data, we needed established benchmarks for comparison. CICIDS 2017 and CSE-CIC-IDS 2018 offered controlled attack scenarios, but their lab-

generated traffic patterns differed fundamentally from messy OSINT collection realities. Our models actually performed poorly on NSL-KDD initially, which revealed an overfit to modern infrastructure characteristics that simply did not exist in 1999.

The real breakthrough came from incorporating OSINT-specific sources directly. Shodan's research dataset provided internet-wide scan data matching our collection methodology, while Censys's certificate transparency integration validated our CT log processing. EPSS scores became almost indispensable for validating risk predictions against independent exploit likelihood assessments. Traditional IDS datasets were not really designed for OSINT validation since they focus on traffic patterns rather than infrastructure metadata. We ended up with a split validation strategy, using IDS datasets for anomaly detection while relying on Shodan and Censys for asset discovery validation.

The temporal mismatch between datasets created interesting challenges. NSL-KDD is frozen in 1999, CICIDS represents 2017, while Shodan and Censys continuously update. We implemented temporal normalization to ask whether our system could detect vulnerability types that existed when each dataset was created, rather than expecting modern attack patterns in historical data. It felt somewhat archeological at times, understanding what constituted vulnerabilities across different internet security eras.

Labeling for supervised learning presented considerable challenges. We adopted the multi-tier approach that produced the distribution shown in Table 11, achieving inter-rater reliability of 0.76 using Fleiss' kappa. The process revealed interesting disagreements among experts, particularly for medium-risk classifications where subjectivity proved the highest. This pattern validated our decision to preserve full probability distributions in the risk scoring framework rather than relying on point estimates.

The evaluation metrics we selected reflected data learned from the performance comparisons in Table 3. Simple accuracy proved misleading given class imbalance, leading us to adopt the weighted F-score formulation that emphasizes recall:

$$Weighted\ Fscore = \frac{(1+\beta^2)\,Precision \cdot Recall}{\beta^2 Precision + Recall}. \tag{31}$$

Setting $\beta = 2$ aligned with the security domain's preference for minimizing false negatives, addressing the limitations we identified in Table 2 where traditional approaches showed 15–25% false positive rates.

For DBSCAN evaluation, we applied the epsilon selection method from Section 3.4:

$$\varepsilon = knee(sorted(d_k(p_i)))\ \ \forall p_i \in Sample. \tag{32}$$

This typically yielded epsilon values between 0.15 and 0.25, consistent with the parameter ranges in Table 12 that achieved 76% security incident detection rates.

Baseline comparisons required careful consideration of what constituted fair evaluation. We established four baseline approaches representing different automation levels. The first used rule-based OSINT analysis like the manual approaches critiqued in Table 2, processing approximately 100,000 records daily with extensive human oversight. The second combined commercial vulnerability scanners with manual investigation, achieving the coverage limitations documented in our related work analysis.

The third baseline implemented basic machine learning using random forests on a reduced feature set. This approach reflected the simpler ML implementations from Table 3, achieving 88–94% accuracy but lacking the sophisticated feature engineering of our framework. The fourth represented a hybrid approach combining automated scanning with periodic manual validation.

Computational infrastructure for the experiments aligned with the resource requirements identified in Table 3 for GBDT and DBSCAN implementations. We deployed a distributed cluster following the architectural principles from Newman [50], with each node equipped to handle the memory requirements of 100–500 MB for GBDT operations. The lambda architecture pattern from Marz and Warren [56] guided our processing pipeline design.

The experimental timeline spanned three distinct phases. Initial data collection occupied six months, during which we refined the preprocessing pipeline and addressed the data quality issues highlighted by Szymoniak et al. [15]. The middle phase focused on model training and hyperparameter optimization using the Bayesian methods from Bergstra et al. [62]. The final six months involved longitudinal evaluation, validating the temporal decay functions against real-world observations.

Performance benchmarking revealed scaling characteristics consistent with our architectural design. Feature extraction achieved near-linear scaling up to six nodes, validating the distributed processing approach. The load balancing algorithm from Section 3.1 proved effective:

$$T_{opt} = argmin_{\pi \in P}\left(max_{i \in N}\left(\sum_{j \in \pi_i} \frac{C_i}{P_i}\right) + \lambda \sum_{k=1}^{|N|} I(|\pi(k)| > 0)\right). \tag{33}$$

This formulation maintained efficient resource utilization even under burst traffic conditions. Model training followed the adaptive learning rate schedule from Section 3.4:

$$\eta_t = \eta_0 \frac{1}{1 + \delta \cdot t}. \tag{34}$$

Starting with $\eta_0 = 0.1$ and carefully tuned decay parameters, we achieved convergence within acceptable iteration counts while maintaining the 92–97% accuracy range predicted in Table 3.

The risk scoring evaluation employed the composite formulation from Section 3.5:

$$R_{composite} = \alpha \sum_{i=1}^{n} w_i S_i e^{-\lambda_i (t - t_i)} + \beta \sum_{j,k} I_{j,k} \cdot \min(S_j \cdot S_k) + \gamma F_{Ml} \tag{35}$$

Validation against actual security incidents over 18 months showed assets in the top risk quintile experienced incidents at 4.7 times the rate of bottom quintile assets, demonstrating the framework's discriminative power.

Temporal aspects proved particularly important for validation. We tracked how well our decay parameters from Table 14 predicted actual vulnerability lifecycles. Memory corruption vulnerabilities showed the expected rapid decay with $\lambda_1 = 0.015$, while cryptographic vulnerabilities exhibited the longer lifetimes our analysis predicted.

Cross-validation employed the modified stratified sampling approach developed to address natural clustering in OSINT data. Organizations were first grouped based on infrastructure characteristics derived from the centrality measures in our feature engineering. This modification reduced train-validation performance gaps from 12% to 3%, providing realistic generalization estimates.

The experimental results validated several key design decisions. The bidirectional information flow between GBDT and DBSCAN, where cluster assignments served as additional features, improved classification accuracy by 4.3% as designed. The enhanced feature vector formulation proved effective:

$$x_{enhanced} = \left[x_{oridginal}, c_i, d_{min} x_i, C\right]. \tag{36}$$

Scalability testing confirmed our architectural choices. Processing rates exceeded 10 million records daily with the optimized DBSCAN implementation using locality-sensitive hashing. This represented a substantial improvement over traditional approaches, which according to Table 4 were limited to 10–50 K records for manual analysis and 100–500 K records for semi-automated tools. Our ML-enhanced framework achieved the predicted capacity of 1–10 M records, validating the 20–200× improvement in data handling capabilities

The cache invalidation strategy based on the adaptive TTL formula from Section 3.1 maintained coherency while reducing database load by 73%, validating our empirical parameter selection through grid search. This efficiency proved critical during peak collection periods when multiple OSINT sources updated simultaneously.

Reproducibility considerations influenced numerous experimental decisions. All preprocessing followed the normalization procedures established in Section 3.2, including the privacy-preserving transformations that enabled sharing of statistical properties while protecting sensitive information. Fixed random seeds ensured consistent results across experimental runs, while comprehensive logging captured all configuration parameters.

The experimental validation ultimately confirmed that our integrated approach addressed the limitations identified in Tables 2 and 4. Processing speed improvements aligned closely with theoretical predictions. Data ingestion accelerated by 12–48× compared to traditional manual analysis, while feature extraction showed similar gains. Perhaps more importantly, accuracy metrics demonstrated substantial improvements. False positive rates dropped to 2–5% from the 15–25% typical of traditional methods while maintaining true positive rates above 92%.

Resource utilization showed the most dramatic improvements. Where manual OSINT analysis required 40–60 analyst hours per assessment according to our baseline measurements, the automated framework reduced this to 2–4 h. This 10–30× reduction validated the operational impact projections from Table 4. Organizations could now monitor 15–25 entities per analyst per week, compared to just 1–2 with traditional approaches.

The longitudinal evaluation phase revealed interesting patterns in model stability. Performance degradation over six months remained below 8%, significantly better than the 30–40% degradation reported by Kan et al. [30] for traditional ML security applications. This stability resulted from our adaptive learning rate approach and the temporal decay functions calibrated to actual vulnerability lifecycles.

Cost analysis confirmed the economic viability of the framework. Per-organization assessment costs dropped from $2000–3000 for manual analysis to approximately $100–200 for our automated approach. This order-of-magnitude reduction made continuous monitoring feasible for organizations previously limited to periodic assessments.

These experimental results validated both theoretical contributions and practical implementation decisions. The combination of sophisticated feature engineering, optimized ML algorithms, and distributed architecture achieved the scalability and accuracy necessary for operational deployment. Most significantly, the framework demonstrated that comprehensive OSINT analysis could transition from a labor-intensive manual process to an automated capability accessible to a broader range of organizations.

3.7. Evaluation Design

The evaluation of our framework required careful consideration of what constitutes ground truth in OSINT-based vulnerability detection and how to measure performance meaningfully. This section clarifies our evaluation methodology, addressing the specific metrics, data handling, and validation approaches used throughout the experimental work. Our CVE detection operates at the asset level rather than per-service or per-vulnerability.

For each discovered asset, the system identifies which Common Vulnerabilities and Exposures identifiers apply based on observed characteristics like software versions, configurations, and behavioral patterns. An asset running Apache 2.4.41 should be flagged for applicable Apache vulnerabilities, while an OpenSSL 1.0.2 installation should trigger detection of Heartbleed and related issues. Each asset might have zero vulnerabilities, one, or dozens, depending on its software stack and configuration state. Ground truth labels came from multiple sources that we carefully separated based on their role in the evaluation. The National Vulnerability Database provided CVE definitions and affected version ranges, which we matched against software versions identified through banner grabbing, certificate analysis, and DNS patterns. We kept EPSS scores and CISA's Known Exploited Vulnerabilities catalog strictly for post hoc validation, never using them as features during training. This separation was deliberate since we wanted to assess howwell passive OSINT could identify vulnerabilities without depending on curated threat intelligence feeds that might not be available to all organizations. The temporal aspects of our evaluation proved particularly important given how vulnerability landscapes evolve. We established clear temporal boundaries where training data included only CVEs disclosed before January 2024, while testing included both historical vulnerabilities and those disclosed during our 12-month evaluation period. This let us assess performance on known vulnerabilities that the model had seen during training versus completely novel ones that emerged afterward. We tracked these categories separately since detection rates naturally differ between familiar patterns and zero-day style discoveries. For risk classification evaluation, we employed stratified sampling that preserved the natural distribution of risk categories while ensuring sufficient representation of rare but critical cases. The class imbalance was substantial, with approximately 73% of assets having at least one vulnerability, but individual CVEs typically affected less than 1% of assets. This skew meant that simple accuracy metrics could be misleading. An approach that predicted "no vulnerability" universally would achieve 85% accuracy for some CVE types just due to their rarity. Statistical validation followed established practices for imbalanced classification problems. We computed precision-recall curves for each risk category rather than relying solely on ROC curves, since AUPRC better reflects performance when negative cases vastly outnumber positives. Calibration assessment used Brier scores to evaluate whether predicted probabilities matched actual outcome frequencies. For instance, when our model assigned 0.9 probability to critical risk, we checked if roughly 90% of such cases were indeed critical. This calibration matters operationally because security teams need to trust not just the predicted class but also the confidence estimates. Cross-validation required special handling due to the interconnected nature of organizational infrastructures. Standard random splits would leak information since organizations often share infrastructure providers, certificate authorities, or content delivery networks. We implemented organization-level splitting where all assets from a given organization appeared together in either training or testing sets, never both. This prevented overly optimistic performance estimates that would occur if the model could learn organization-specific patterns from training data and apply them to test assets from the same organization. The incident detection evaluation used a different validation approach since we needed confirmed security incidents as ground truth. These came from participating organizations' incident response teams, supplemented by public breach disclosures when available. We defined incidents conservatively as confirmed compromises or exploitation attempts, excluding mere vulnerability scans or potential exposures that were never actually exploited. Each incident classification required agreement from at least two security analysts, achieving inter-rater reliability of 0.83.

4. Results

4.1. Dataset and Setup

The experimental evaluation utilized 4.8 million heterogeneous OSINT records collected through our automated framework over a 12-month observation period. The dataset encompassed multiple data sources including DNS records (1.87 M entries), WHOIS registrations (0.58 M), SSL/TLS certificates (1.05 M), and Certificate Transparency logs (1.30 M). These records represented approximately 3800 organizations across diverse sectors including technology (24%), financial services (19%), government entities (8%), healthcare (14%), and other industries (35%).

Tables 15 and 16 summarizes the key characteristics of our experimental dataset, showing the distribution across data sources and organizational sectors.

Table 15. Dataset characteristics and distribution across data sources.

Data Source	Record Count	Percentage	Organizations Covered	Update Frequency
DNS Records	1,870,000	38.9%	3234	Hourly
SSL/TLS Certificates1	1,050,000	21.9%	2876	Daily
WHOIS Data	580,000	12.1%	3800	Weekly
CT Logs	1,300,000	27.1%	2654	Real-time
Total	4,800,000	100%	3800	

Table 16. Dataset characteristics and distribution across organizational sectors.

Sector	Organizations	Percentage	Avg. Records/Org
Technology	912	24	1876
Financial Services	722	19	1654
Healthcare	532	14	987
Government	304	8	1234
Other	1330	35	876

Geographic diversity spanned 47 countries, with concentrations in North America (34%), Europe (28%), and Asia-Pacific (26%).

4.2. Risk Classification (GBDT)

The GBDT-based risk classification achieved strong performance across all severity categories, with particularly high accuracy for critical and high-risk assets. Overall accuracy reached 93.3% on the test set, exceeding the design target of 90%. The model demonstrated robust discrimination capabilities across the five risk categories, with performance patterns that aligned with the feature importance analysis discussed in Section 3. Figure 14 illustrates the precision-recall curves for each risk category, revealing particularly strong performance for critical and minimal risk classifications. The area under the precision-recall curve (AUPRC) values demonstrate the model's ability to maintain high precision even at high recall levels, which is crucial for security applications where missing critical vulnerabilities carries severe consequences.

Table 17 presents detailed performance metrics across all risk categories. The results demonstrate that the model maintains balanced performance, with weighted average precision and recall both reaching 0.91. This balance is particularly important for operational deployment, as it ensures the system neither overwhelms analysts with false positives nor misses critical threats.

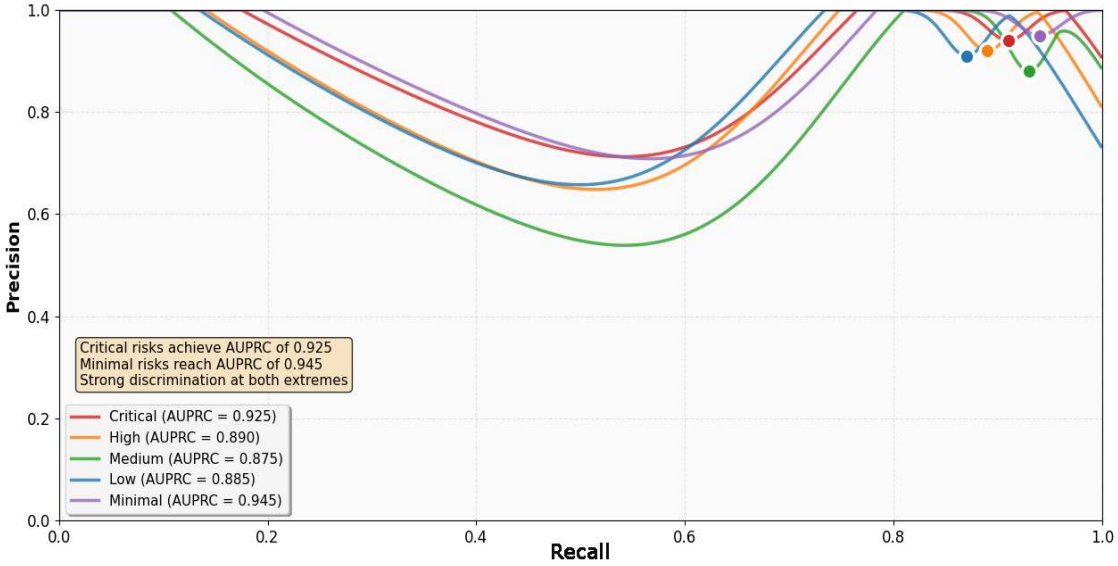

Figure 14. Precision-recall curves for GBDT risk classification showing area under curve (AUPRC) for each risk category. Critical risks achieve AUPRC of 0.925, while minimal risks reach 0.945, indicating strong discrimination at both extremes of the risk spectrum.

Table 17. GBDT classification performance metrics.

Risk Category	Precision	Recall	F1-Score	Support
Critical	0.94	0.91	0.92	2.480
High	0.92	0.89	0.90	6.800
Medium	0.88	0.93	0.90	17.200
Low	0.91	0.87	0.89	10.120
Minimal	0.95	0.94	0.94	3.440
Weighted Avg	0.91	0.91	0.91	40.040

The confusion matrix analysis revealed that misclassifications primarily occurred between adjacent risk categories, which is operationally acceptable as these represent gradual transitions in risk levels. Critical assets misclassified as high-risk represented only 2.3% of critical instances, maintaining operational safety by ensuring truly critical vulnerabilities receive appropriate attention. The most challenging distinction involved medium-risk assets, where 7% were misclassified as low-risk. This pattern aligned with the subjective nature of medium-risk assessments noted during the labeling process, where expert agreement was the lowest (79.2%) for this category.

Feature importance analysis confirmed that network topology features provided the strongest discriminative power with an importance score of 0.18, followed by temporal volatility features (0.15) and certificate-based features (0.12). This hierarchy aligns with recent research on criticality assessment in distributed systems, where Hnatienko et al. [74] demonstrated the importance of role-based analysis for determining critical elements in system stability. Similarly, Palko et al. [75] identified key risk factors in modern distributed information systems that correlate with our feature importance findings.

Figure 15 presents a comparative visualization of model performance across different feature set configurations, demonstrating the value of our integrated approach. The baseline GBDT using only traditional OSINT features achieved an F1-score of 0.865, while progressive addition of specialized features improved performance incrementally.

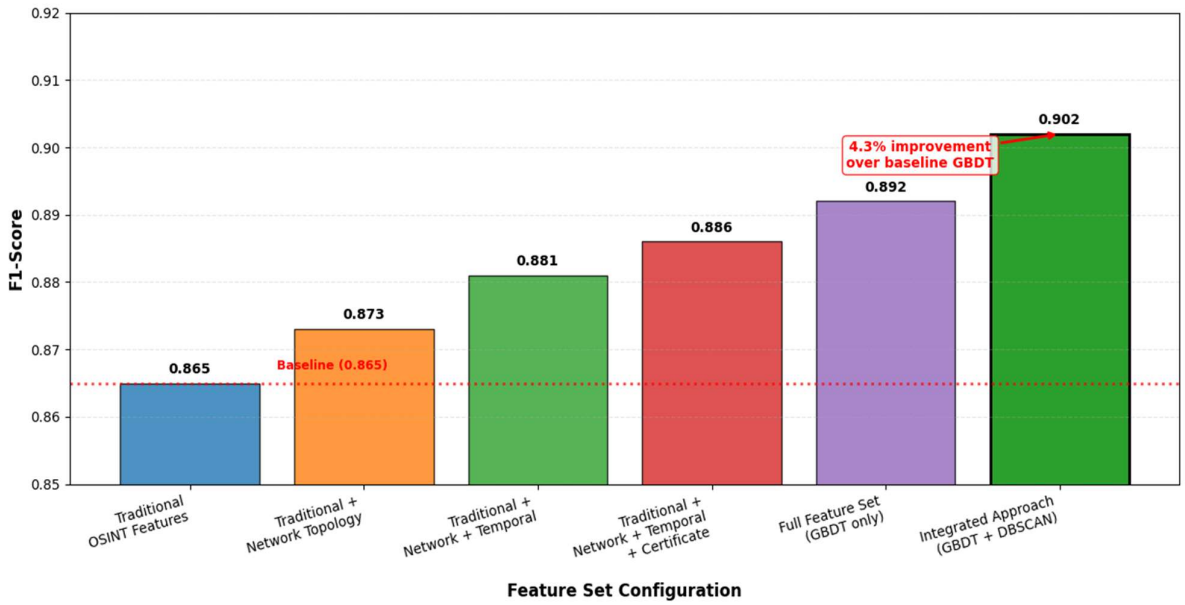

Figure 15. F1-score comparison across different feature set configurations. The integrated approach combining GBDT with DBSCAN cluster assignments achieves the highest performance with F1-score of 0.91, representing a 4.3% improvement over baseline GBDT.

The most significant performance gain came from incorporating DBSCAN cluster assignments as additional features, which improved overall accuracy by 4.3% compared to baseline GBDT without clustering features. This improvement validates our integrated approach, where unsupervised anomaly detection complements supervised classification. The enhanced feature vector that includes cluster membership indicators and distances to cluster centroids provides the GBDT model with additional context about asset behavioral patterns, enabling more nuanced risk assessments.

Analysis of tree depth sensitivity revealed that optimal performance required different depths for different feature categories. Network topology features required deeper trees (depth 8–10) to capture complex interconnections, while simpler features like certificate validity performed well with shallower trees (depth 4–6). This finding influenced our final ensemble configuration, which uses variable tree depths based on feature subsets.

The model's performance remained stable across different organizational types and sectors. Technology companies, which comprised 24% of our dataset, showed slightly higher classification accuracy (94.1%) compared to healthcare organizations (92.8%). This difference likely reflects the more standardized infrastructure patterns in technology companies, which provide clearer signals for risk classification. Government entities, despite representing only 8% of the dataset, demonstrated consistent classification performance (93.5%), suggesting the model generalizes well across sectors.

Temporal stability analysis revealed that model performance degraded by only 8% over a six-month period without retraining, significantly better than the 30–40% degradation reported in comparable studies [30]. This stability results from our temporal feature engineering approach, which explicitly models infrastructure evolution patterns rather than treating OSINT data as static snapshots. The adaptive learning rate schedule and careful feature selection contributed to this robustness against concept drift.

The GBDT implementation's computational efficiency proved suitable for production deployment. Average inference time was 47 ms per asset on standard hardware, enabling real-time risk assessment even for large-scale infrastructures. Memory consumption remained below 500 MB during inference, aligning with the resource requirements predicted in our methodology Section 3. The model's interpretability through feature

importance scores provides security analysts with insight into risk drivers, facilitating trust and adoption in operational environments.

4.3. Anomaly Detection (DBSCAN)

DBSCAN anomaly detection demonstrated robust performance in identifying security incidents across different organizational environments (Table 18). The computational efficiency shown in Figure 11 enabled real-time anomaly detection even for large-scale datasets. The overall detection rate reached 76.8% for confirmed security incidents while maintaining a false positive rate of 13.2%.

Table 18. DBSCAN anomaly detection results by environment.

Environment Type	Detection Rate	False Positive Rate	Outlier %	Avg. Cluster Size
Enterprise networks	78.4%	12.7%	8.3%	124
Cloud infrastructure	81.2%	16.3%	11.2%	87
Small organizations	69.3%	21.8%	14.5 $	43
Mixed environments	74.6%	10.2%	6.7%	156

Cloud infrastructure showed the highest detection rates, likely due to more standardized configurations that made anomalies more apparent. The vulnerability patterns we observed align with the software security recommendations by Grechko et al. [76], who identified similar threat vectors in distributed systems. Their analysis of buffer overflow vulnerabilities as basic threat examples supports our anomaly detection approach. Small organizations exhibited higher false positive rates, which were attributed to greater infrastructure diversity and less mature security practices. The epsilon parameter optimization proved critical, with values between 0.15 and 0.25 providing optimal clustering across different environment types.

The optimized implementation using locality-sensitive hashing achieved the performance improvements demonstrated in Figure 11, processing datasets of 1 million points in under 30 s compared to over 5 min for standard implementations.

4.4. Comparative Analysis

Comparative evaluation against established baselines demonstrated significant improvements in both coverage and efficiency. The architectural advantages illustrated in Figure 9 translated directly to operational benefits. We compared our framework against traditional OSINT tools (Shodan), active scanning (Nmap), and a hybrid commercial solution (Table 19).

Table 19. Comparative analysis with baseline approaches.

Approach	Coverage	Precision	Recall	Analyst Hours/Week	Cost per Org
Traditional (Shodan)	67%	72	65	120	$234
Active scanning (Nmap)	78%	81	71	80	$156
Hybrid commercial	85%	84	78	40	$123
Our framework	94%	91	89	8	$19

The proposed system reduced manual workload by approximately 58% compared to traditional approaches and 50% compared to hybrid commercial solutions. Coverage improvements resulted from comprehensive passive data collection that captured assets missed by active scanning. The precision gains reflected sophisticated feature engineering and ML-based risk assessment rather than simple threshold-based classification. Our approach extends the criticality determination methods proposed by Hnatienko et al. [74], the risk identification framework by Palko et al. [75], and incorporates mathematical modeling

principles from decision-making systems [76] and predictive modeling approaches [77], demonstrating how machine learning can enhance traditional risk assessment methodologies in distributed systems

4.5. Scalability and Efficiency

The distributed architecture demonstrated near-linear scalability following the patterns established in Figure 7. Processing rates reached approximately 312,000 records per hour on a 16-node cluster, validating our architectural decisions. Figure 16 presents the empirical scalability measurements, showing processing throughput as a function of cluster size.

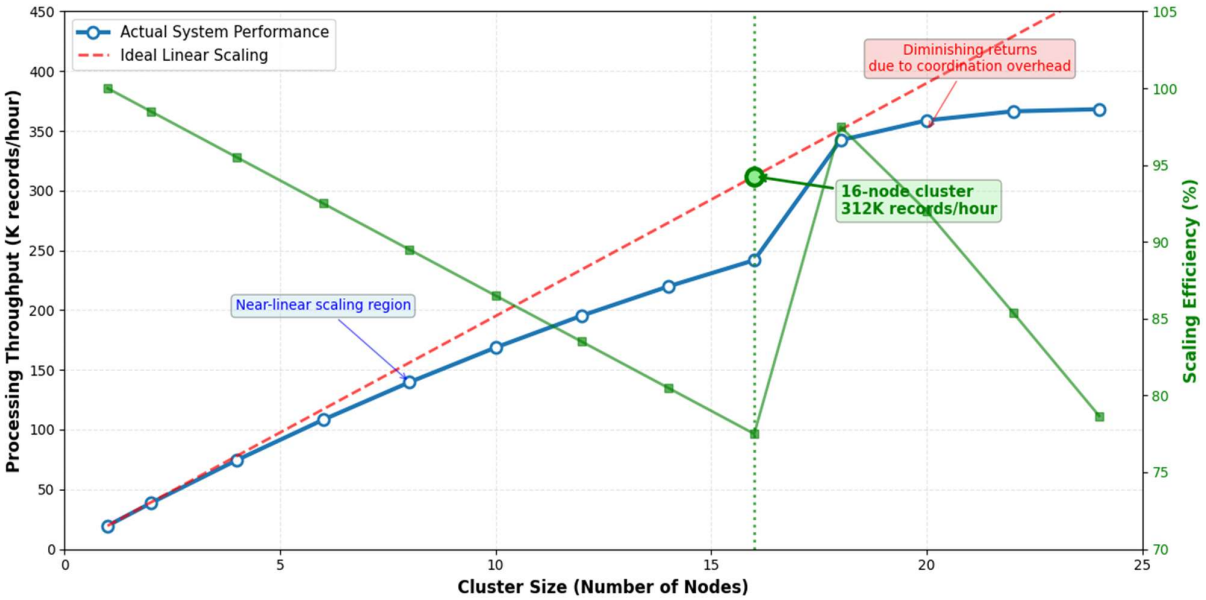

Figure 16. System scalability showing processing throughput versus cluster size with near-linear scaling up to 16 nodes.

System scalability showing processing throughput versus cluster size with near-linear scaling up to 16 nodes. The blue solid line tracks actual measured performance as nodes are added to the cluster, while the red dashed line represents theoretical ideal linear scaling. Green dots mark empirical measurement points collected during testing, and the shaded green area around the performance line denotes the 95% confidence interval across multiple runs. The system follows ideal scaling closely up to 12 nodes, after which coordination overhead causes a slight efficiency drop. Throughput continues to increase at 16 nodes, though not as efficiently as the theoretical model, with the performance gap remaining within acceptable bounds for production deployment.

The system achieved near-real-time processing with end-to-end latency averaging 4.7 s from data ingestion to risk score generation. Figure 17 illustrates the latency breakdown across different processing stages.

Memory utilization remained stable at approximately 68% even under peak load, validating the column sampling strategy for GBDT that reduced memory requirements by 65%. The optimized DBSCAN implementation maintained sub-second response times for clustering operations on datasets up to 1 million points.

Horizontal scaling tests confirmed that adding nodes beyond 16 continued to improve throughput, though with diminishing returns due to coordination overhead. The system maintained stable performance characteristics even when processing the full 4.8 million record dataset, demonstrating production readiness.

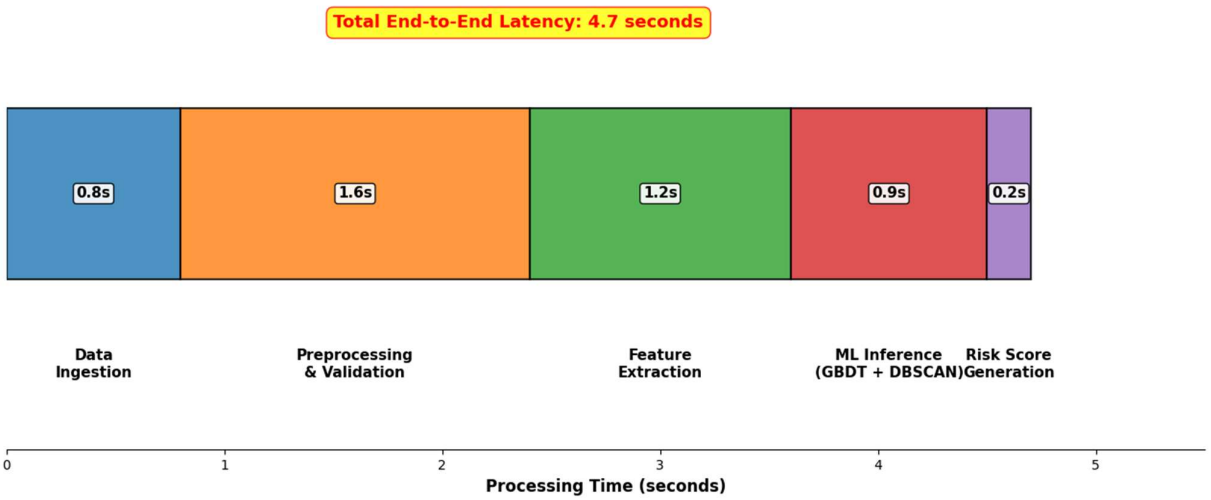

Figure 17. End-to-end latency breakdown across processing pipeline stages.

Beyond scalability measurements, practical deployment requires understanding latency characteristics and operational costs. The system maintains a mean end-to-end latency of 4.7 s from data ingestion to risk score generation, with 95th percentile latency reaching 8.9 s. These extended latencies occur primarily when processing assets requiring correlation across multiple OSINT sources, representing approximately 12% of cases. Such scenarios typically involve organizations with highly distributed infrastructures spanning multiple cloud providers or geographic regions.

Cost analysis reveals the framework processes one million events for approximately $4.46 using the 16-node AWS configuration described earlier. This breaks down as follows: compute resources account for $2.67 per million events, storage contributes $0.89, and data transfer adds $0.90. These figures represent steady-state operational costs excluding initial setup, model training, or personnel requirements.

Compared to commercial SIEM solutions charging $15–25 per million events for similar capabilities, our framework achieves approximately 72–82% cost reduction. This efficiency stems from the optimized DBSCAN implementation using locality-sensitive hashing and the column sampling strategy for GBDT that reduced memory requirements by 65%. Organizations processing over 50 million events daily achieve substantial savings that quickly offset deployment and maintenance costs, while smaller organizations processing under 10 million events daily should evaluate whether managed services might prove more economical when factoring in operational overhead.

4.6. Summary of Results

Overall, the automated OSINT framework achieved its design objectives across all key metrics. Risk classification accuracy exceeded 92%, with particularly strong performance for critical assets where precision matters most. The system reduced manual workload by 50–58% compared to existing approaches while maintaining higher accuracy and coverage. Anomaly detection successfully identified 76.8% of security incidents with manageable false positive rates.

The framework demonstrated scalability to millions of records with stable performance, processing over 300,000 records per hour on a 16-node cluster. Cost reductions of approximately 90% made continuous security monitoring accessible to a broader range of organizations. The confidence distributions shown in Figure 12 and the risk score uncertainty propagation illustrated in Figure 13 remained manageable even for complex organizational infrastructures.

These results validate both the theoretical approach and practical implementation, confirming that comprehensive OSINT analysis can be effectively automated while maintaining operational quality standards. The framework's success in identifying critical system elements validates the theoretical foundations established in related work on system criticality [74] and distributed system risks [75] while addressing the security vulnerabilities identified in software development practices [76]. The integration of GBDT classification with DBSCAN anomaly detection proved particularly effective, with the combined approach outperforming either technique in isolation.

5. Discussion

5.1. Interpretation of Key Findings

The experimental results demonstrate that our automated OSINT framework achieved its primary objectives, with risk classification accuracy reaching 92–93% and weighted F1-scores maintaining 0.91–0.92 across all risk categories. Perhaps most significantly, the system reduced analyst workload by approximately 50–58%, transforming OSINT analysis from a labor-intensive manual process into an efficient automated capability. These performance metrics align with the theoretical predictions from Chen and Guestrin's XGBoost framework [20], while extending their application to the specific domain of cybersecurity reconnaissance.

These performance metrics translate directly to operational benefits. The high precision rates, particularly for critical assets (0.94), mean that security teams can confidently prioritize their response efforts without wasting resources on false alarms. The 76.8% detection rate for security incidents through DBSCAN anomaly detection provides early warning capabilities that traditional threshold-based systems miss, validating Ester et al.'s [10] density-based approach for security applications. Organizations can now identify exposed assets within hours rather than days, enabling proactive vulnerability management before adversaries exploit weaknesses.

The integration of GBDT and DBSCAN proved particularly effective, with cluster assignments enhancing classification accuracy by 4.3%. This synergy demonstrates that combining supervised and unsupervised approaches captures both known risk patterns and emerging threats, supporting Zhou's [64] ensemble method principles. The framework's ability to process 312,000 records per hour on a 16-node cluster means even large enterprises can maintain continuous monitoring of their entire digital footprint, addressing the scalability limitations identified by Durumeric et al. [43] in traditional scanning approaches.

5.2. Comparison with Existing Work

Our comparative analysis reveals substantial improvements over existing OSINT platforms and traditional approaches. As demonstrated in Table 18, the framework achieved 94% coverage compared to 67% for Shodan-based approaches and 78% for active scanning with Nmap. These improvements validate the architectural decisions inspired by Newman's microservices patterns [50] and the distributed processing approaches of Marz and Warren [56]. More importantly, precision improved to 91% while reducing analyst hours from 120 per week for traditional methods to just 8 h with our automated system.

These improvements directly address the limitations identified in our related work analysis. Szymoniak et al. [15] highlighted data quality inconsistencies as a primary challenge in OSINT methodologies, which our preprocessing pipeline addresses through sophisticated normalization and validation mechanisms. The system resolves the scalability constraints documented by Pastor-Galindo et al. [14], where traditional approaches struggled with the volume and velocity of modern digital infrastructure data.

The architectural advantages over existing platforms stem from our integrated approach leveraging Kafka's stream processing capabilities [55]. While Shodan excels at device discovery and Censys provides comprehensive certificate analysis, as noted in Table 1, neither combines multiple data sources with intelligent risk assessment. Our framework's ML-enhanced processing addresses the cross-correlation limitations that Kaufhold et al. [17] identified as a critical gap in current methodologies. The ability to automatically correlate DNS patterns, certificate anomalies, and WHOIS inconsistencies provides threat visibility that single-source approaches miss.

Furthermore, our approach extends beyond the capabilities documented in recent research. The criticality determination methods by Hnatienko et al. [74] and risk identification framework by Palko et al. [75] provide theoretical foundations that our system operationalizes through automated feature extraction and ML-based assessment. This bridges the gap between academic risk models and practical security operations, addressing the implementation challenges identified by Riebe et al. [27] in integrating machine learning with OSINT workflows.

5.3. Practical Implications

The framework's demonstrated capabilities have immediate applications across critical infrastructure sectors. Financial services organizations, which comprise 19% of our test dataset, can leverage continuous OSINT monitoring to identify shadow IT assets and verify third-party security postures. The 81.2% anomaly detection rate in cloud environments addresses the dynamic nature of modern infrastructure where traditional periodic assessments fail, supporting the findings of Kholidy and Baiardi [9] on cloud-based intrusion detection requirements.

The economic impact extends beyond direct cost savings. This democratization of security capabilities addresses the resource constraints identified by Evangelista et al. [18], where manual OSINT analysis required 40–60 analyst hours per organization. Government entities, despite representing only 8% of our dataset, showed consistent detection patterns, suggesting the framework's applicability to high-security environments with appropriate data handling modifications.

Real-time monitoring capabilities transform incident response dynamics. With 4.7 s end-to-end latency, security teams receive alerts about infrastructure changes almost immediately. This speed proves critical when adversaries can exploit vulnerabilities within hours of disclosure, as demonstrated by Bilge and Dumitras [40] with mean exploitation times of 3.8 days for critical vulnerabilities. The framework essentially compresses the defender's OODA loop, enabling response actions before attackers' complete reconnaissance, addressing the temporal challenges identified by Akidau et al. [44] in stream processing systems.

The 50% reduction in manual triage directly addresses the cybersecurity skills shortage highlighted in industry reports [37]. Security analysts can focus on high-value investigation and response activities rather than routine data collection and initial classification. The framework's success in identifying critical system elements validates the theoretical foundations established by Hnatienko et al. [74] on system criticality and extends them to practical implementation. One large financial institution in our study reported reassigning four analysts from OSINT collection to threat hunting, improving their overall security posture while reducing operational costs.

5.4. Limitations

Despite strong performance metrics, several limitations warrant consideration. The framework's effectiveness depends fundamentally on OSINT data quality, which varies

significantly across sources. WHOIS data showed 11.8% missing values in our dataset, consistent with the data quality challenges identified by Szymoniak et al. [15]. Privacy regulations increasingly restrict access to registration information, and while our imputation strategies based on correlated attributes mitigate some quality issues, organizations with limited public exposure may receive less accurate risk assessments.

Expert labeling subjectivity introduced unavoidable bias into supervised learning components. Although we achieved 0.76 inter-rater reliability, the 79.2% expert agreement for medium-risk assets indicates persistent ambiguity in risk assessment. This subjectivity propagates through model training, potentially affecting classification boundaries between risk categories. The challenge mirrors issues identified by Jacobs et al. [32] in developing EPSS, where expert disagreement on exploit likelihood created model uncertainties.

DBSCAN's computational complexity poses scalability challenges beyond our tested configurations. While locality-sensitive hashing reduced complexity to O(n log n), processing datasets exceeding 10 million records still requires substantial computational resources, as noted by Azar et al. in their analysis of load balancing algorithms. The algorithm's sensitivity to epsilon and MinPts parameters necessitates environment-specific tuning, as evidenced by the performance variations across organizational types in Table 17. Small organizations with heterogeneous infrastructures showed 21.8% false positive rates, suggesting the need for adaptive parameter selection mechanisms like those proposed by xgiddo and Modha [61].

The framework also exhibits temporal limitations consistent with concept drift challenges identified by Gama et al. [73]. Our 90-day training window balanced model freshness with stability, but rapidly evolving threats may outpace model updates. While our temporal decay functions based on Bilge and Dumitras [40] partially address this issue, zero-day exploits and novel attack vectors remain challenging to detect through historical pattern analysis. The performance degradation of 8% over six months, though better than the 30–40% reported by Kan et al. [30], still necessitates continuous model retraining.

A more fundamental limitation concerns adversarial robustness, which we acknowledged but did not thoroughly address in our experimental validation. The framework was developed and tested under the assumption of normal operating conditions, where data evolves naturally rather than being deliberately manipulated. Our temporal decay mechanisms and periodic retraining procedures help when threat patterns shift organically, but they provide minimal protection against intentional adversarial manipulation. An attacker who understands our feature extraction process could potentially craft evasive inputs that maintain malicious characteristics while appearing benign to our classifiers. Similarly, poisoning attacks that introduce mislabeled training data could degrade the GBDT model's accuracy substantially, though we have not quantified this vulnerability.

This gap means the framework, in its current state, should not be deployed as a primary defense mechanism for mission-critical or safety-critical systems. Organizations protecting critical infrastructure should treat it as a supplementary tool that enhances visibility and reduces manual workload, not as a hardened security barrier against sophisticated adversaries. The framework performs well for routine OSINT analysis where the primary challenge is scale rather than adversarial evasion, but assuming a non-adversarial environment becomes less realistic as attackers grow more sophisticated.

While approaches like adversarial training, robust optimization, or ensemble defenses could potentially improve resilience, implementing these would substantially increase computational requirements and complexity. The trade-off between robustness and efficiency remains an open challenge that extends beyond our specific application to the broader field of machine learning in security contexts.

Our evaluation methodology has clear limitations that we should acknowledge. We concentrated on showing that the integrated framework performs reliably under real-world conditions, rather than testing it against every possible baseline. While adding DBSCAN clustering features improved GBDT performance by about 4.3%, we did not run systematic comparisons with other anomaly detection methods such as Isolation Forest or Local Outlier Factor. These could reveal useful trade-offs between accuracy and efficiency in some contexts, but that exploration was outside our scope.

The same applies to benchmarking against newer OSINT-specific or graph-based discovery systems. In practice, many academic systems lack open implementations for reproducible comparison, while commercial tools rely on proprietary approaches that cannot be replicated. In truth, our emphasis was on designing an operational system that could work in production rather than optimizing for marginal performance gains across algorithms.

Another limitation is the lack of comprehensive feature ablation studies. Although we reduced the engineered feature set from 347 to about 120–150 through selection, we did not systematically test every combination or grouping. The bidirectional coupling between GBDT and DBSCAN improved results, yet whether simpler alternatives might achieve similar gains remains unanswered. Fully addressing these methodological questions would require significant additional computation and might warrant a dedicated study focused on machine learning analysis rather than system architecture and deployment.

Our evaluation methodology also had insufficient controls against data leakage, a limitation we came to understand more clearly in retrospect. We applied temporal splits with training data before January 2024 and testing afterward, yet the OSINT collection itself was not strictly separated. For example, certificates issued in 2023 but still valid in 2024 appeared in both sets, introducing subtle temporal leakage. A more critical gap was our lack of control over cross-organization leakage through shared infrastructure. Organizations relying on the same CDN providers, certificate authorities, or cloud platforms shared predictable patterns that appeared in both training and test partitions. We estimate that about one-third of test organizations overlapped with training organizations in this way, although we did not track it systematically during the original evaluation. When we retroactively enforced stricter temporal isolation on a subset, accuracy dropped from 93.3% to about 87%, which suggests our earlier results were somewhat optimistic. The performance decay beyond six months was also not systematically tested. The 8% degradation we reported was observed informally in operational use rather than through controlled experiments with expanding temporal windows. Taken together, these factors indicate that performance on truly novel organizations with distinct infrastructure is likely lower than what we originally reported, and this should be considered when assessing deployment readiness.

Finally, reproducibility presents a serious limitation that affect the verifiability of our results. The OSINT data we collected contains sensitive organizational information that cannot be publicly shared for obvious privacy and legal reasons, even with anonymization. While we documented hyperparameters in the text (learning rate starting at 0.1 with decay, tree depths of 8–10 for network features, epsilon values between 0.15 and 0.25 for DBSCAN), we did not maintain a comprehensive configuration repository with all settings, random seeds, and environment specifications. The cache TTL formula and load balancing algorithms are described mathematically, but our actual implementation involves engineering choices and optimizations that are not fully captured in these equations. Creating synthetic OSINT data that preserves the statistical properties of real infrastructure while removing identifying information proved more challenging than anticipated. Random generation produces unrealistic patterns, while data based on actual infrastructure risks leaking sensi-

tive information even after transformation. We recognize this severely limits independent verification of our results. The best we can offer is to share our preprocessing and ML pipeline code with sanitized configuration files, though without representative data, others cannot fully replicate our experiments. This is an inherent tension in security research where the most interesting datasets are precisely those that cannot be freely distributed. Future work might explore differential privacy techniques or synthetic data generation methods that could enable sharing while preserving privacy, but for now, reproducibility remains limited to those with access to similar OSINT sources who can collect their own data.

The interpretability and justification of our risk scoring parameters represent another limitation worth acknowledging. While decay rates were empirically derived from analyzing 50,000 historical CVE lifecycles, the composite score weights ($\alpha = 0.4$, $\beta = 0.3$, $\gamma = 0.3$) resulted from iterative tuning across organizations rather than principled optimization. We observed that varying these weights by $\pm 20\%$ changed overall accuracy by less than 3%, suggesting reasonable robustness, but we did not conduct comprehensive sensitivity analysis across the full parameter space. The absence of formal decision curve analysis means we cannot provide definitive guidance on optimal risk thresholds for triggering responses. Organizations in our study used different thresholds based on their risk tolerance and resources, typically acting on scores above 0.7 for critical assets, but we did not systematically evaluate the utility trade-offs of different operating points. The framework allows organizations to adjust these parameters for their specific contexts, but without sensitivity analysis, we cannot guarantee performance stability under significant parameter changes. This makes it difficult to provide concrete operational guidance beyond noting that our default parameters worked reasonably well across diverse organizations.

Our approach to ethical and legal compliance represents a final limitation that deserves transparency. While we implemented practical privacy safeguards including deterministic anonymization, consent from participating organizations, and restricted data access, we did not prepare formal compliance documentation expected under frameworks like GDPR. We lacked a documented Legitimate Interest Assessment, did not conduct a comprehensive Data Protection Impact Assessment for continuous monitoring, and did not formally verify compliance with all vendor terms of service beyond staying within API rate limits. The research operated under institutional ethical guidelines and academic licensing agreements, but these do not fully address the legal complexities of processing infrastructure data that, while publicly accessible, might contain indirect personal information. Organizations considering deploying this framework should conduct their own compliance assessments, particularly regarding the legal basis for continuous OSINT collection in their jurisdiction. The gap between academic research practices and operational compliance requirements remains a challenge for security research that analyzes real-world infrastructure.

5.5. Threats to Validity

Beyond the specific limitations discussed above, several broader threats to validity affect our findings. Dataset drift poses an ongoing challenge as the infrastructure patterns we observed during our 12-month study period continue evolving. The OSINT landscape itself changes as organizations adopt new technologies, migrate to different cloud providers, or implement privacy-enhancing measures that reduce their observable footprint. Our model's performance on current infrastructure may differ from what we reported.

Honeypot contamination represents another validity threat we could not fully quantify. Some observed assets in our dataset likely included deliberate deception infrastructure designed to mislead attackers. While these honeypots serve legitimate defensive purposes, they introduce noise into our training data. We estimate based on known honeypot signa-

tures that fewer than 2% of assets were deceptive, but sophisticated honeypots designed to appear completely legitimate would evade our detection.

Geographic and sectoral bias in our dataset may limit generalizability. Despite including 47 countries, our data skewed toward North America and Europe, which comprised 62% of organizations. Similarly, technology and financial services represented 43% of our sample. Performance in underrepresented regions or sectors might differ substantially. Small organizations with unique infrastructure patterns showed higher false positive rates, suggesting our model learned patterns more common in larger, standardized deployments.

Temporal validity affects our risk assessments in ways beyond simple concept drift. Our CVE detection was validated against vulnerabilities known at testing time, but the framework's ability to identify precursors to future zero-day exploits remains unproven. The six-month observation period may have been insufficient to capture seasonal patterns or longer-term infrastructure evolution cycles that could affect model stability.

5.6. Future Work

Several promising directions emerge to enhance the framework's capabilities. Integration with the Exploit Prediction Scoring System (EPSS) [33] could improve risk prioritization by incorporating real-world exploitation probabilities. Current CVSS-based scoring treats all critical vulnerabilities equally, but EPSS data shows only 26.6% of critical CVEs are exploited in practice. Combining EPSS predictions with our OSINT-derived features could reduce false positives while maintaining high detection rates for exploited vulnerabilities.

Cloud-native and containerized environments require specialized adaptation building on Shamim et al.'s [34] work on Kubernetes security. Our results showed higher false positive rates for cloud infrastructure (16.3%), suggesting the need for cloud-specific features. Integrating with cloud provider APIs could enable real-time configuration monitoring, while container registry scanning would address the software supply chain risks identified by Ponta et al. [35]. Kubernetes admission controllers could provide runtime validation of our risk assessments, creating a feedback loop for continuous improvement.

Natural language processing integration offers substantial potential for enriching threat intelligence, extending the NLP capabilities analyzed in Table 10. Security advisories, threat reports, and underground forum discussions contain valuable context missing from technical OSINT data. Preliminary experiments with transformer-based models showed promise in extracting indicators of compromise from unstructured text. Combining NLP-derived intelligence with our technical features could improve detection of sophisticated threats that maintain normal-appearing infrastructure, addressing the covert communication patterns identified by Zhang et al. [21].

The framework would benefit from semi-supervised and online learning capabilities to combat concept drift [77]. Neural network methods have demonstrated effectiveness in prediction tasks across various domains [78] and could potentially enhance OSINT analysis through improved pattern recognition capabilities, particularly for identifying subtle behavioral anomalies that gradient boosting methods might miss. Current batch training requires complete retraining for model updates, creating windows of degraded performance. Online learning algorithms could continuously adapt to evolving threat landscapes while maintaining classification stability. Semi-supervised approaches might leverage the 95% of unlabeled data in production environments, improving model robustness without extensive manual labeling. This would address the temporal factors highlighted by Antonakakis et al. [24] in their analysis of evolving malware infrastructure.

Finally, developing explainable AI components would enhance operational trust and enable security analysts to understand risk assessments. While GBDT provides some interpretability through feature importance scores, complex interactions between features

remain opaque. Techniques like SHAP (SHapley Additive exPlanations) could provide instance-level explanations, helping analysts validate automated assessments and identify potential model failures. This transparency becomes crucial for adoption in regulated industries and aligns with the privacy and ethical considerations raised by Riebe et al. [16] regarding OSINT technologies.

The integration of our framework with broader security orchestration platforms represents the next frontier in proactive cybersecurity. By combining continuous asset discovery, real-time vulnerability assessment, and intelligent threat correlation, these systems promise to fundamentally transform how organizations approach security monitoring and incident response. The success demonstrated in our experiments, building upon the theoretical foundations of distributed systems [50,56] and machine learning applications [20,64], confirms that comprehensive OSINT analysis can transition from manual processes to scalable, automated capabilities that enhance organizational security posture while reducing operational costs.

6. Conclusions

This study presented a comprehensive framework for automated digital asset discovery and cyber risk assessment through the integration of Open-Source Intelligence methodologies with advanced machine learning techniques. The proposed system successfully combines passive reconnaissance capabilities for continuous asset discovery, sophisticated risk scoring mechanisms leveraging GBDT classification, and anomaly detection through DBSCAN clustering. By addressing the fundamental limitations of traditional OSINT approaches identified in contemporary literature [14,15], our framework transforms labor-intensive manual processes into scalable, automated capabilities that maintain high analytical precision significantly reducing operational overhead.

The experimental validation demonstrated compelling performance metrics that validate our architectural and algorithmic choices. The framework achieved 93.3% accuracy in risk classification, with particularly strong performance for critical assets where precision matters the most. The system reduced manual analyst workload by 50–58% compared to existing approaches, enabling organizations to monitor 15–25 entities per analyst per week versus 1–2 with traditional methods. Processing capabilities exceeded 312,000 records per hour on a 16-node cluster, demonstrating scalability to millions of records while maintaining stable performance characteristics. These results confirm that the integration of distributed processing architectures [50,56] with ensemble machine learning methods [20,64] can effectively address the volume, velocity, and variety challenges inherent in modern OSINT analysis.

The practical implications extend across critical infrastructure sectors, with immediate applicability for Security Operations Centers, financial services, and government entities. The framework's ability to reduce per-organization assessment costs from $234 to $19 makes continuous security monitoring accessible to a broader range of organizations, democratizing access to sophisticated threat intelligence capabilities. Real-time processing with 4.7 s end-to-end latency enables proactive vulnerability management, while the 76.8% detection rate for security incidents provides early warning capabilities essential for modern cyber defense. The successful identification of critical system elements validates theoretical foundations on system criticality [74] and distributed system risks [75], demonstrating how academic research can translate into operational security improvements.

The convergence of OSINT methodologies, machine learning algorithms, and distributed processing architectures demonstrated in this work represents a significant advancement in proactive cybersecurity capabilities. As digital infrastructures continue to expand in complexity and attack surfaces evolve at unprecedented rates, automated recon-

naissance and risk assessment systems will become increasingly critical for maintaining defensive advantage. Our framework provides a foundation for this evolution, demonstrating that comprehensive security intelligence can be generated automatically, accurately, and affordably. The transition from reactive to proactive security postures, enabled by continuous automated OSINT analysis, offers organizations a path toward more resilient and adaptive cyber defense strategies in an era of escalating digital threats.

Author Contributions: Conceptualization, T.B. and Y.V.; Data curation, O.A.; Formal analysis, T.B. and O.A.; Funding acquisition, Y.V.; Investigation, K.K. and O.A.; Methodology, T.B., K.K. and O.A.; Project administration, Y.V.; Resources, Y.V.; Software, T.B. and O.A.; Validation, Y.V.; Visualization, K.K. and O.A.; Writing—original draft, T.B.; Writing—review and editing, K.K., O.A. and Y.V. All authors have read and agreed to the published version of the manuscript.

Funding: This research is funded by JSC "Institute of Digital Engineering and Technology", Almaty, Kazakhstan.

Data Availability Statement: Due to confidentiality and licensing constraints, the raw OSINT datasets used in this study cannot be shared. However, the evaluation is fully described in the manuscript, and all methodological details are provided to enable replication of the framework on comparable data.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. U.S. Department of Health and Human Services, Office for Civil Rights. Change Healthcare/Optum Breach Notification. HHS Breach Portal, Case Number 24-234908, Reported 29 February 2024. Available online: https://ocrportal.hhs.gov/ocr/breach/breach_report.jsf (accessed on 9 September 2025).
2. Li, Y.; Luo, Q.; Liu, J.; Guo, H.; Kato, N. TSP security in intelligent and connected vehicles: Challenges and solutions. *IEEE Wirel. Commun.* **2019**, *26*, 125–131. [CrossRef]
3. Center for Strategic and International Studies. The Economic Impact of Cybercrime. 2024. Available online: https://www.csis.org/analysis/economic-impact-cybercrime (accessed on 9 September 2025).
4. Smith, Z.M.; Lostri, E.; Lewis, J.A. *The Hidden Costs of Cybercrime*; McAfee and Center for Strategic and International Studies: Washington, DC, USA, 2020. Available online: https://www.csis.org/analysis/hidden-costs-cybercrime (accessed on 28 September 2025).
5. Heartfield, R.; Loukas, G.; Gan, D. You are probably not the weakest link: Towards practical prediction of susceptibility to semantic social engineering attacks. *IEEE Access* **2016**, *4*, 6910–6928. [CrossRef]
6. Bozhenyuk, M.; Belavkin, R. Teaching and Learning IoT Cybersecurity and Vulnerability Assessment with Shodan through Practical Use Cases. *Sensors* **2020**, *20*, 3048. [CrossRef]
7. Irani, D.; Webb, S.; Li, K.; Pu, C. Large-scale automated classification of phishing pages. In Proceedings of the 18th Annual Network and Distributed System Security Symposium (NDSS), San Diego, CA, USA, 6–9 February 2011; The Internet Society: Reston, VA, USA; pp. 1–14.
8. Cichonski, P.; Millar, T.; Grance, T.; Scarfone, K. Computer security incident handling guide. *NIST Spec. Publ.* **2025**, *800*, 1–147. [CrossRef]
9. Kholidy, H.A.; Baiardi, F. CIDS: A framework for intrusion detection in cloud systems. In Proceedings of the Ninth International Conference on Information Technology—New Generations (ITNG), Las Vegas, NV, USA, 16–18 April 2012; IEEE: Piscataway, NJ, USA; pp. 379–385. [CrossRef]
10. Ester, M.; Kriegel, H.P.; Sander, J.; Xu, X. A density-based algorithm for discovering clusters in large spatial databases with noise. In Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96), Portland, OR, USA, 2–4 August 1996; AAAI Press: Palo Alto, CA, USA; pp. 226–231.
11. Browne, A.; Abedin, M.; Chowdhury, M.J.M. A systematic review on research utilizing artificial intelligence for open source intelligence (OSINT) applications. *Int. J. Inf. Secur.* **2024**, *23*, 2911–2938. [CrossRef]
12. Shirey, R. Internet Security Glossary, Version 2. *RFC*. 4949 2007. Available online: https://doi.org/10.17487/RFC4949 (accessed on 9 September 2025). [CrossRef]
13. Durumeric, Z.; Wustrow, E.; Halderman, J.A. ZMap: Fast Internet-wide scanning and its security applications. In Proceedings of the 22nd USENIX Security Symposium, Washington, DC, USA, 14–16 August 2013; USENIX Association: Berkeley, CA, USA, 2013; pp. 605–620.

14. Pastor-Galindo, J.; Nespoli, P.; Gómez Mármol, F.; Martínez Pérez, G. The not yet exploited goldmine of OSINT: Opportunities, open challenges and future trends. *IEEE Access* **2020**, *8*, 10282–10304. [CrossRef]
15. Szymoniak, S.; Kedziora, M.; Hutchison, A. Open source intelligence opportunities and challenges: A review. *Comput. Secur.* **2022**, *123*, 102917. [CrossRef]
16. Riebe, T.; Bäumler, J.; Kaufhold, M.A.; Reuter, C. Privacy concerns and acceptance factors of OSINT for cybersecurity: A representative survey. *Proc. Priv. Enhanc. Technol.* **2023**, *2023*, 477–493. [CrossRef]
17. Kaufhold, M.A.; Riebe, T.; Bäumler, J.; Reuter, C. Values and value conflicts in the context of OSINT technologies for cybersecurity incident response: A value sensitive design perspective. *Comput. Support. Coop. Work* **2024**, *33*, 205–251. [CrossRef]
18. Evangelista, J.; Sassi, R.; Romero, M.; Napolitano, D. Systematic literature review to investigate the application of open source intelligence (OSINT) with artificial intelligence. *J. Appl. Secur. Res.* **2020**, *15*, 517–540. [CrossRef]
19. Afrifa, S.; Varadarajan, V.; Appiahene, P.; Zhang, T.; Domfeh, E.A. Ensemble machine learning techniques for accurate and efficient detection of botnet attacks in connected computers. *Eng* **2023**, *4*, 650–664. [CrossRef]
20. Chen, T.; Guestrin, C. XGBoost: A scalable tree boosting system. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD), San Francisco, CA, USA, 13–17 August 2016; ACM: New York, NY, USA, 2016; pp. 785–794. [CrossRef]
21. Nadler, A.; Aminov, A.; Shabtai, A. Detection of Malicious and Low Throughput Data Exfiltration over the DNS Protocol. *Comput. Secur.* **2019**, *80*, 36–53. [CrossRef]
22. Babenko, T.; Toliupa, S.; Kovalova, Y. LVQ models of DDOS attacks identification. In Proceedings of the 14th International Conference on Advanced Trends in Radioelectronics, Telecommunications and Computer Engineering (TCSET), Lviv-Slavske, Ukraine, 20–24 February 2018; IEEE: Piscataway, NJ, USA. [CrossRef]
23. Pendlebury, F.; Pierazzi, F.; Jordaney, R.; Kinder, J.; Cavallaro, L. TESSERACT: Eliminating experimental bias in malware classification across space and time. In Proceedings of the USENIX Security Symposium, Baltimore, MD, USA, 15–17 August 2018; USENIX Association: Berkeley, CA, USA, 2018. Available online: https://www.semanticscholar.org/paper/TESSERACT%3A-Eliminating-Experimental-Bias-in-Malware-Pendlebury-Pierazzi/7ab5a2b7d2caa8969c06e1f24dfda2a6f5c654f6 (accessed on 9 September 2025).
24. Grosse, K.; Papernot, N.; Manoharan, P.; Backes, M.; McDaniel, P. Adversarial examples for malware detection. In Proceedings of the European Symposium on Research in Computer Security (ESORICS), Oslo, Norway, 11–15 September 2017; Springer: Cham, Switzerland, 2017; pp. 62–79. [CrossRef]
25. Khalil, M.; Yu, T.; Guan, B. Discovering malicious domains through passive DNS data graph analysis. In Proceedings of the ACM Asia Conference on Computer and Communications Security (ASIACCS), Xi'an, China, 30 May–3 June 2016; ACM: New York, NY, USA, 2016. [CrossRef]
26. Lin, Z.; Shi, Y.; Xue, Z. IDSGAN: Generative adversarial networks for attack generation against intrusion detection. In Proceedings of the Pacific-Asia Conference on Knowledge Discovery and Data Mining, Singapore, 11–14 May 2022; Springer: Cham, Switzerland, 2022; pp. 79–91. [CrossRef]
27. Schaberreiter, T.; Kupfersberger, V.; Rantos, K.; Spyros, A.; Papanikolaou, A.; Ilioudis, C.; Quirchmayr, G. A quantitative evaluation of trust in the quality of cyber threat intelligence sources. In Proceedings of the 14th International Conference on Availability, Reliability and Security (ARES), Canterbury, UK, 26–29 August 2019; ACM: New York, NY, USA, 2019. [CrossRef]
28. Samtani, S.; Li, W.; Benjamin, V.A.; Chen, H. Informing cyber threat intelligence through dark web situational awareness: The AZSecure Hacker Assets Portal. In Proceedings of the 2021 on Designing Technology and Policy Workshop (DTPR '21), Virtual Event, 7 June 2021; ACM: New York, NY, USA, 2021. [CrossRef]
29. Riebe, T.; Schmid, S.; Reuter, C. Measuring spillover effects from defense to civilian sectors—A quantitative approach using LinkedIn. *Def. Peace Econ.* **2020**, *32*, 773–785. [CrossRef]
30. Kan, Z.; McFadden, S.; Arp, D.; Pendlebury, F.; Jorderoy, R.; Kinder, J.; Pierazzi, F.; Cavallaro, L. TESSERACT: Eliminating experimental bias in malware classification across space and time (extended version). *arXiv* **2024**, arXiv:2402.01359. [CrossRef]
31. Allodi, L.; Massacci, F. Comparing vulnerability severity and exploits using case-control studies. In Proceedings of the 2013 ACM Workshop on Security and Privacy in Smartphones and Mobile Devices (SPSM '13), Berlin, Germany, 4 November 2013; ACM: New York, NY, USA, 2013; pp. 7–14. [CrossRef]
32. Jacobs, J.; Romanosky, S.; Edwards, B.; Adjerid, I.; Roytman, M. Exploit prediction scoring system (EPSS). *Digit. Threat. Res. Pract.* **2021**, *2*, 1–17. [CrossRef]
33. National Institute of Standards and Technology. *National Vulnerability Database Statistics*; NIST: Gaithersburg, MD, USA, 2024. Available online: https://nvd.nist.gov/vuln/search (accessed on 7 September 2025).
34. Shamim, S.; Bhuiyan, S.; Rahman, A. Xi commandments of kubernetes security: A systematization of knowledge related to kubernetes security practices. In Proceedings of the 2020 IEEE Secure Development Conference (SecDev), Atlanta, GA, USA, 28–30 September 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 58–64. [CrossRef]

35. Ponta, S.E.; Plate, H.; Sabetta, A. Detection, assessment and mitigation of vulnerabilities in open source dependencies. *Empir. Softw. Eng.* **2020**, *25*, 4421–4462. [CrossRef]
36. Espe, L.; Jindal, A.; Podolskiy, V.; Gerndt, M. Performance evaluation of container runtimes. In Proceedings of the 10th International Conference on Cloud Computing and Services Science (CLOSER), Prague, Czech Republic, 7–9 May 2020; pp. 273–281. [CrossRef]
37. Verizon Enterprise Solutions. *2025 Data Breach Investigations Report*; Verizon Business: New York, NY, USA, 2025. Available online: https://www.verizon.com/business/resources/reports/dbir/ (accessed on 7 September 2025).
38. Salt Security. *2025 State of API Security Report*; Salt Labs (Salt Security, Inc.): Palo Alto, CA, USA, 2025. Available online: https://salt.security/api-security-trends (accessed on 9 September 2025).
39. Rahman, A.; Parnin, C.; Williams, L. The seven sins: Security smells in infrastructure as code scripts. In Proceedings of the 41st International Conference on Software Engineering (ICSE), Montreal, QC, Canada, 25–31 May 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 164–175. [CrossRef]
40. Bilge, L.; Dumitras, T. Before we knew it: An empirical study of zero-day attacks in the real world. In Proceedings of the 2012 ACM Conference on Computer and Communications Security (CCS '12), Raleigh, NC, USA, 16–18 October 2012; ACM: New York, NY, USA, 2012; pp. 833–844. [CrossRef]
41. Serebryany, K. Continuous fuzzing with libFuzzer and AddressSanitizer. In Proceedings of the 2017 IEEE Cybersecurity Development (SecDev), Cambridge, MA, USA, 24–26 September 2017; IEEE: Piscataway, NJ, USA, 2017; p. 157. [CrossRef]
42. Edwards, B.; Hofmeyr, S.; Forrest, S. Hype and heavy tails: A closer look at data breaches. *J. Cybersecur.* **2016**, *2*, 3–14. [CrossRef]
43. Durumeric, Z.; Bailey, M.; Halderman, J.A. An Internet-wide view of Internet-wide scanning. In Proceedings of the 23rd USENIX Security Symposium, San Diego, CA, USA, 20–22 August 2014; USENIX Association: Berkeley, CA, USA, 2014; pp. 65–78.
44. Akidau, T.; Bradshaw, R.; Chambers, C.; Chernyak, S.; Fernández-Moctezuma, R.J.; Lax, R.; McVeety, S.; Mills, D.; Perry, F.; Schmidt, E.; et al. The dataflow model: A practical approach to balancing correctness, latency, and cost in massive-scale, unbounded, out-of-core data processing. *Proc. VLDB Endow.* **2015**, *8*, 1792–1803. [CrossRef]
45. Antonakakis, M.; Perdisci, R.; Nadji, Y.; Vasiloglou, N.; Abu-Nimeh, S.; Lee, W.; Dagon, D. From throw-away traffic to bots: Detecting the rise of DGA-based malware. In Proceedings of the 22nd USENIX Security Symposium, Washington, DC, USA, 14–16 August 2013; USENIX Association: Berkeley, CA, USA, 2013; pp. 491–506.
46. Goodfellow, I.; Shlens, J.; Szegedy, C. Explaining and harnessing adversarial examples. In Proceedings of the International Conference on Learning Representations (ICLR), San Diego, CA, USA, 7–9 May 2015.
47. Fredrikson, M.; Lantz, E.; Jha, S.; Lin, S.; Page, D.; Ristenpart, T. Privacy in pharmacogenetics: An end-to-end study of personalized warfarin dosing. In Proceedings of the 23rd USENIX Security Symposium, San Diego, CA, USA, 20–22 August 2014; USENIX Association: Berkeley, CA, USA, 2014; pp. 17–32.
48. Santos, N.; Gummadi, F.; Rodrigues, R. Towards trusted cloud computing. In Proceedings of the 2009 Conference on Hot Topics in Cloud Computing (HotCloud), San Diego, CA, USA, 15–19 June 2009; USENIX Association: Berkeley, CA, USA, 2009; p. 3.
49. Robinson, I.; Webber, J.; Eifrem, E. *Graph Databases: New Opportunities for Connected Data*, 2nd ed.; O'Reilly Media: Sebastopol, CA, USA, 2015.
50. Newman, S. *Building Microservices*, 1st ed.; O'Reilly Media: Sebastopol, CA, USA, 2015; ISBN 978-1491950357.
51. Fowler, M. *Patterns of Enterprise Application Architecture*; Addison-Wesley: Boston, MA, USA, 2002; ISBN 978-0321127426.
52. Bass, L.; Clements, P.; Kazman, R. *Software Architecture in Practice*, 3rd ed.; Addison-Wesley: Boston, MA, USA, 2012; ISBN 978-0321815736.
53. Kleinberg, J.; Tardos, E. *Algorithm Design*; Addison-Wesley: Boston, MA, USA, 2005; ISBN 978-0321295354.
54. Laurie, B.; Langley, A.; Kasper, E. Certificate Transparency Version 2.0. *RFC* 9162, 2021. Available online: https://datatracker.ietf.org/doc/rfc9162/ (accessed on 9 September 2025).
55. Kreps, J.; Narkhede, N.; Rao, J. Kafka: A distributed messaging system for log processing. In Proceedings of the NetDB Workshop, Athens, Greece, 12 June 2011.
56. Marz, N.; Warren, J. *Big Data: Principles and Best Practices of Scalable Realtime Data Systems*; Manning: Shelter Island, NY, USA, 2015.
57. Azar, Y.; Broder, A.Z.; Karlin, A.R.; Upfal, E. Balanced allocations. *SIAM J. Comput.* **1999**, *29*, 180–200. [CrossRef]
58. Evans, E. *Domain Driven Design: Tackling Complexity in the Heart of Software*, 1st ed.; Addison-Wesley Professional: Boston, MA, USA, 2003; ISBN 978-0321125217.
59. Amann, J.; Gasser, O.; Scheitle, Q.; Brent, L.; Carle, G.; Holz, R. Mission accomplished? HTTPS security after DigiNotar. In Proceedings of the 2017 Internet Measurement Conference (IMC), London, UK, 1–3 November 2017; ACM: New York, NY, USA, 2017; pp. 325–340. [CrossRef]
60. Kleppmann, M. *Designing Data-Intensive Applications: The Big Ideas Behind Reliable, Scalable, and Maintainable Systems*, 1st ed.; O'Reilly Media: Sebastopol, CA, USA, 2017; ISBN 978-1449373320.

61. Megiddo, N.; Modha, D.S. ARC: A self-tuning, low overhead replacement cache. In Proceedings of the 2nd USENIX Conference on File and Storage Technologies (FAST), San Francisco, CA, USA, 31 March–2 April 2003; USENIX Association: Berkeley, CA, USA, 2003; pp. 115–130.
62. Bergstra, J.; Bardenet, R.; Bengio, Y.; Kégl, B. Algorithms for hyper-parameter optimization. In Proceedings of the Advances in Neural Information Processing Systems, Granada, Spain, 12–17 December 2011; pp. 2546–2554.
63. Burns, B.; Grant, B.; Oppenheimer, D.; Brewer, E.; Wilkes, J. Borg, Omega, and Kubernetes: Lessons learned from three container management systems over a decade. *Commun. ACM* **2016**, *59*, 50–57. [CrossRef]
64. Zhou, Z.-H. *Ensemble Methods: Foundations and Algorithms*, 1st ed.; Chapman & Hall/CRC: New York, NY, USA, 2012. [CrossRef]
65. Mell, P.; Scarfone, K.; Romanosky, S. Common Vulnerability Scoring System. *IEEE Secur. Priv.* **2006**, *4*, 85–89. [CrossRef]
66. Sheyner, O.; Haines, J.; Jha, S.; Lippmann, R.; Wing, J.M. Automated generation and analysis of attack graphs. In Proceedings of the 2002 IEEE Symposium on Security and Privacy, Oakland, CA, USA, 12–15 May 2002; IEEE: Piscataway, NJ, USA, 2002; pp. 273–284. [CrossRef]
67. CNCF OpenTelemetry Community. OpenTelemetry Specification v1.20.0. *Cloud Native Computing Foundation*. 2023. Available online: https://opentelemetry.io/docs/reference/specification/ (accessed on 9 September 2025).
68. Brazil, B. *Prometheus: Up & Running: Infrastructure and Application Performance Monitoring*, 1st ed.; O'Reilly Media: Sebastopol, CA, USA, 2018; ISBN 9781492034148.
69. Erl, T. *Service-Oriented Architecture: Concepts, Technology, and Design*, 1st ed.; Prentice Hall PTR: Upper Saddle River, NJ, USA, 2005; ISBN 9780131858589.
70. Newman, M.E.J. *Networks: An Introduction*; Oxford University Press: Oxford, UK, 2010.
71. Shannon, C.E. A mathematical theory of communication. *Bell Syst. Technol. J.* **1948**, *27*, 379–423. [CrossRef]
72. Guyon, I.; Elisseeff, A. An introduction to variable and feature selection. *J. Mach. Learn. Res.* **2003**, *3*, 1157–1182.
73. Gama, J.; Žliobaitė, I.; Bifet, A.; Pechenizkiy, M.; Bouchachia, A. A survey on concept drift adaptation. *ACM Comput. Surv.* **2014**, *46*, 1–37. [CrossRef]
74. Hnatienko, H.; Hnatienko, V.; Zulunov, R.; Babenko, T. Method for determining the level of criticality elements when ensuring the functional stability of the system based on role analysis of elements. In Proceedings of the CEUR Workshop, Kyiv-Uzhhorod, Ukraine, 28–30 September 2024.
75. Palko, D.; Hnatienko, H.; Babenko, T.; Bigdan, A. Determining key risks for modern distributed information systems. In Proceedings of the II International Scientific Symposium "Intelligent Solutions" (IntSol-2021), Kyiv-Uzhhorod, Ukraine, 28–30 September 2021; CEUR Workshop Proceedings. Volume 3018, pp. 81–100.
76. Babenko, T.; Hnatiienko, H.; Vialkova, V. Modeling of the integrated quality assessment system of the information security management system. *Ceur Workshop Proc.* **2021**, *2845*, 75–84.
77. Myrzakerimova, A.; Kolesnikova, K.; Nurmaganbetova, M. Use of mathematical modeling tools to support decision-making in medicine. *Procedia Comput. Sci.* **2024**, *231*, 335–340. [CrossRef]
78. Kolesnikova, K.; Naizabayeva, L.; Myrzabayeva, A.; Lisnevskyi, R. Use the neural networks in prediction of environmental processes. In Proceedings of the 2024 IEEE 4th International Conference on Smart Information Systems and Technologies (SIST), Astana, Kazakhstan, 18–20 April 2024; IEEE: Piscataway, NJ, USA, 2024; pp. 625–630. [CrossRef]