International Journal of Advanced Research in Computer Science & Technology (IJARCST)

| ISSN: 2347-8446 | www.ijarcst.org | editor@ijarcst.org |A Bimonthly, Peer Reviewed & Scholarly Journal|

||Volume 9, Issue 5, September-October 2026||

DOI:10.15662/IJARCST.2026.0905004

Explainable AI for Automated DevSecOps Security Validation across Enterprise Cloud Infrastructure Platforms

Dr. Fu-Chien Kao

Associate Professor, Department of Computer Science and Information Engineering, Da-Yeh University, Taiwan

Publication History: Received: 25.08.2026; Revised: 01.10.2026; Accepted: 04.10. 2026; Published: 10.10.2026.

ABSTRACT: The rapid adoption of enterprise cloud infrastructure has increased the complexity of cybersecurity management, creating a need for automated, scalable, and transparent security validation mechanisms. DevSecOps integrates security practices into the software development and deployment lifecycle; however, traditional validation approaches often struggle to identify sophisticated vulnerabilities across distributed cloud environments. Artificial intelligence (AI) offers opportunities to improve vulnerability detection, risk prioritization, configuration assessment, and compliance monitoring. Nevertheless, the limited interpretability of many AI-driven security systems presents challenges for enterprise adoption, particularly when security decisions require accountability, auditability, and regulatory justification. This research investigates the application of Explainable Artificial Intelligence (XAI) to automated DevSecOps security validation across enterprise cloud infrastructure platforms. The proposed approach integrates machine learning-based threat detection, infrastructure configuration analysis, policy compliance assessment, and explainability techniques into continuous integration and continuous deployment pipelines. Explainability methods, including feature importance analysis, local interpretable explanations, and SHapley Additive exPlanations, are considered for clarifying security alerts and risk predictions. The research methodology combines a conceptual framework, experimental evaluation, and comparative analysis using representative cloud security scenarios. Evaluation criteria include detection accuracy, false-positive rate, validation latency, explanation consistency, and operational scalability. The study aims to establish how explainable AI can improve security decision-making, strengthen trust in automated validation, and support continuous compliance across heterogeneous enterprise cloud environments while preserving deployment efficiency and human oversight.

KEYWORDS: Explainable Artificial Intelligence, DevSecOps, Automated Security Validation, Enterprise Cloud Infrastructure, Cloud Security, Machine Learning, Continuous Integration and Continuous Deployment, Vulnerability Detection, Security Compliance, SHAP, LIME, Risk Assessment, Infrastructure as Code, Continuous Security Monitoring

I. INTRODUCTION

The increasing adoption of cloud computing has transformed enterprise information technology by enabling scalable infrastructure, flexible resource allocation, distributed application deployment, and cost-efficient service delivery. Organizations increasingly rely on public, private, and hybrid cloud environments to support business-critical applications, data processing, and digital services. However, the complexity of these environments introduces significant security challenges, including misconfigured resources, excessive access privileges, vulnerable software dependencies, exposed application programming interfaces, insecure network configurations, and inconsistent compliance controls. Traditional security validation methods, which frequently depend on manual reviews and periodic assessments, may not adequately address the speed and complexity of modern software delivery. Consequently, organizations require security mechanisms capable of continuously evaluating infrastructure, detecting emerging vulnerabilities, and enforcing security policies throughout the development and deployment lifecycle. DevSecOps addresses this requirement by integrating security controls into development, operations, and deployment workflows rather than treating security as a separate activity performed after software development. Through automated testing, dependency scanning, infrastructure validation, container analysis, and continuous monitoring, DevSecOps enables organizations to identify security weaknesses earlier and reduce the likelihood of vulnerabilities reaching production environments. Nevertheless, implementing DevSecOps across large enterprise cloud infrastructures remains challenging because different platforms employ distinct security configurations, access-control mechanisms,



| ISSN: 2347-8446 | www.ijarcst.org | editor@ijarcst.org |A Bimonthly, Peer Reviewed & Scholarly Journal|

||Volume 9, Issue 5, September-October 2026||

DOI:10.15662/IJARCST.2026.0905004

monitoring services, and compliance requirements. These differences increase the difficulty of establishing consistent security validation practices across heterogeneous infrastructure environments.

Artificial Intelligence (AI) and machine learning provide additional opportunities to improve automated DevSecOps security validation by analyzing large volumes of security events, identifying anomalous configurations, correlating vulnerability indicators, and prioritizing risks according to contextual characteristics. Machine learning models can support the identification of suspicious activities that conventional rule-based systems may overlook, while automated reasoning mechanisms can help determine whether infrastructure configurations violate organizational security policies. Despite these advantages, many AI-driven security systems operate as complex predictive models whose decisions are difficult for security engineers and compliance auditors to interpret. An unexplained security alert may indicate a genuine threat, a configuration inconsistency, or a false positive, yet practitioners need sufficient evidence to understand the alert, assess its severity, and determine an appropriate remediation strategy. Explainable Artificial Intelligence (XAI) addresses this limitation by providing interpretable information about the factors contributing to model predictions and automated recommendations. Integrating XAI into DevSecOps can therefore improve transparency, support human verification, strengthen accountability, and facilitate security auditing across enterprise cloud platforms. The central research problem concerns how explainable AI can be integrated into automated security validation without introducing excessive computational overhead or disrupting continuous delivery workflows. This research investigates a conceptual framework that combines cloud configuration analysis, vulnerability assessment, machine learning-based risk classification, explainable decision support, and policy-driven remediation. It seeks to evaluate the effectiveness of explainable AI in improving detection performance, reducing false positives, increasing the interpretability of security findings, and supporting consistent security governance across diverse enterprise cloud environments.

II. LITERATURE REVIEW

DevSecOps has emerged as an important extension of DevOps, emphasizing the integration of security controls throughout the software development lifecycle. Existing research on secure software development highlights the importance of shifting security activities toward earlier development stages, commonly described as the shift-left approach. Static application security testing, dynamic application security testing, software composition analysis, container image scanning, and infrastructure-as-code validation are widely used to identify weaknesses before deployment. These practices support continuous security assessment and help organizations detect vulnerabilities before they become operational risks. Cloud security research further emphasizes the importance of identity and access management, network segmentation, encryption, configuration governance, and continuous compliance monitoring. Infrastructure-as-code technologies introduce opportunities to validate cloud configurations automatically through policy-as-code mechanisms and repeatable security checks. However, conventional rule-based approaches depend heavily on predefined policies and known vulnerability patterns. They may generate excessive alerts when rules are applied without contextual information, and they can struggle to identify previously unseen attack patterns or complex relationships between cloud resources. Enterprise environments also contain heterogeneous cloud services, multiple deployment pipelines, and changing operational conditions, making unified security validation difficult. These limitations motivate the investigation of intelligent security validation mechanisms that combine conventional policy enforcement with adaptive analytical capabilities.

Machine learning has been increasingly investigated for cybersecurity applications, including intrusion detection, malware classification, anomaly detection, vulnerability prioritization, and security event correlation. Supervised learning algorithms can classify known categories of security weaknesses using labeled datasets, while unsupervised and semi-supervised techniques can identify deviations from established behavioral patterns. Natural language processing can also support the analysis of vulnerability descriptions, security advisories, and configuration documentation. Within DevSecOps, these capabilities may improve the prioritization of vulnerabilities by considering contextual factors such as exploitability, asset criticality, exposure, and potential business impact. Nevertheless, predictive performance alone is insufficient for enterprise security adoption. Black-box models may provide risk scores without explaining which configuration properties, behavioral indicators, or contextual variables influenced their conclusions. This limitation complicates incident investigation, security governance, and regulatory auditing, particularly when automated systems recommend blocking a deployment or modifying production infrastructure. Explainable AI techniques offer approaches for addressing this challenge. SHapley Additive exPlanations (SHAP) estimate feature contributions to model predictions, whereas Local Interpretable Model-agnostic Explanations (LIME) approximate model behavior around individual predictions. Feature importance analysis and interpretable surrogate

| ISSN: 2347-8446 | www.ijarcst.org | editor@ijarcst.org |A Bimonthly, Peer Reviewed & Scholarly Journal|

||Volume 9, Issue 5, September-October 2026||

DOI:10.15662/IJARCST.2026.0905004

models can provide additional insight into decision-making processes. However, explanations may vary across models and datasets, and feature attribution does not necessarily establish causality. Furthermore, generating explanations can introduce computational costs or create misleading confidence if the underlying model is poorly calibrated.

Although existing research provides substantial foundations for cloud security automation, machine learning-based threat detection, and interpretable predictive systems, their integration into a unified enterprise DevSecOps validation architecture remains an important research challenge. Security validation requires more than identifying potentially malicious events; it must connect predictions to actionable evidence, established policies, infrastructure dependencies, and remediation procedures. An effective system should explain why a resource is considered insecure, identify the relevant policy or vulnerability, quantify the associated risk, and recommend an appropriate response while preserving human oversight for high-impact decisions. Cross-platform deployment introduces additional challenges because cloud providers differ in their resource models, security interfaces, logging mechanisms, and compliance implementations. Moreover, explanations must remain understandable to different stakeholders, including developers, cloud engineers, security analysts, and compliance auditors. A useful research framework should therefore evaluate not only detection accuracy but also false-positive rates, processing latency, explanation fidelity, explanation stability, remediation effectiveness, and scalability. It should additionally consider data privacy, model drift, adversarial manipulation, and the possibility that incorrect AI recommendations could interrupt critical enterprise services. This research addresses these concerns by proposing an explainable AI-based approach to automated DevSecOps security validation that combines conventional security controls with contextual risk analysis and interpretable decision support. The framework is intended to establish a foundation for assessing the technical effectiveness and operational usefulness of explainable security automation across heterogeneous enterprise cloud infrastructure platforms.

III. RESEARCH METHODOLOGY

This research adopts a mixed-method, experimental, and comparative methodology to investigate the effectiveness of Explainable Artificial Intelligence in automated DevSecOps security validation across enterprise cloud infrastructure platforms. The methodology combines quantitative measurements of security detection and operational performance with qualitative assessments of explanation clarity, usefulness, and suitability for enterprise security workflows. The research is organized around a conceptual architecture that integrates cloud infrastructure monitoring, security data collection, vulnerability identification, machine learning-based risk classification, explainability mechanisms, and automated validation within continuous integration and continuous deployment pipelines. The proposed architecture is designed to accommodate representative public-cloud, private-cloud, and hybrid-cloud environments without assuming that all platforms expose identical security interfaces. Data sources include infrastructure-as-code templates, cloud configuration records, identity and access policies, container image scan results, software dependency inventories, vulnerability advisories, security event logs, and compliance policy outcomes. Synthetic records and controlled test environments may supplement appropriately authorized operational data to represent different security conditions, including insecure storage configurations, excessive privileges, exposed network services, vulnerable dependencies, and deviations from organizational baselines. Each observation is associated, where feasible, with a validated security label, a relevant policy requirement, or an expert-reviewed assessment. Data collection follows principles of least privilege, confidentiality, data minimization, and secure storage. Sensitive identifiers and production secrets are excluded or anonymized before analysis. The study defines research questions concerning whether explainable AI improves the interpretability of automated security decisions, whether contextual machine learning improves risk prioritization compared with conventional rules, and whether the additional computational cost of generating explanations remains acceptable within continuous delivery workflows. The experimental design establishes baseline systems and evaluation criteria before model training and testing to support reproducibility and reduce the risk of selectively reporting favorable results.

The second methodological component involves data preparation, feature engineering, and the development of predictive models for security validation. Raw security data are normalized into a common analytical representation while retaining essential relationships between cloud resources, software components, configuration properties, and policy requirements. Feature engineering considers characteristics such as public accessibility, privilege scope, encryption status, network exposure, vulnerability severity, exploitability indicators, asset criticality, dependency relationships, configuration deviations, and relevant historical security events. Categorical attributes are encoded using appropriate techniques, numerical features are scaled when required, and missing values are handled according to their meaning and frequency. Data quality procedures identify duplicate records, inconsistent labels, invalid observations, and potential sources of information leakage. Where records are related by application, infrastructure environment, or time, the training and testing partitions are designed to prevent closely related observations from appearing in both sets

International Journal of Advanced Research in Computer Science & Technology (IJARCST)

| ISSN: 2347-8446 | www.ijarcst.org | editor@ijarcst.org |A Bimonthly, Peer Reviewed & Scholarly Journal|

||Volume 9, Issue 5, September-October 2026||

DOI:10.15662/IJARCST.2026.0905004

in ways that would artificially inflate performance. Temporal evaluation may be used to assess how well a model performs on security observations collected after its training period. The study considers several candidate algorithms, including logistic regression as an interpretable baseline, decision trees and random forests for structured security data, and gradient-boosted tree models for potentially complex relationships. Anomaly detection methods may be evaluated separately when reliable labels are unavailable or when identifying previously unseen configuration patterns is important. Hyperparameters are selected using training and validation data, while a held-out test dataset is reserved for final performance assessment. Class imbalance is addressed through suitable class weighting, resampling within training partitions, or threshold adjustment, with care taken to avoid contaminating the evaluation data. Model selection considers not only predictive accuracy but also calibration, robustness, inference time, and compatibility with automated deployment pipelines. The resulting models generate risk classifications or scores that can be combined with deterministic policy checks, ensuring that explicitly prohibited configurations are not treated as acceptable merely because a statistical model assigns them a low predicted risk.

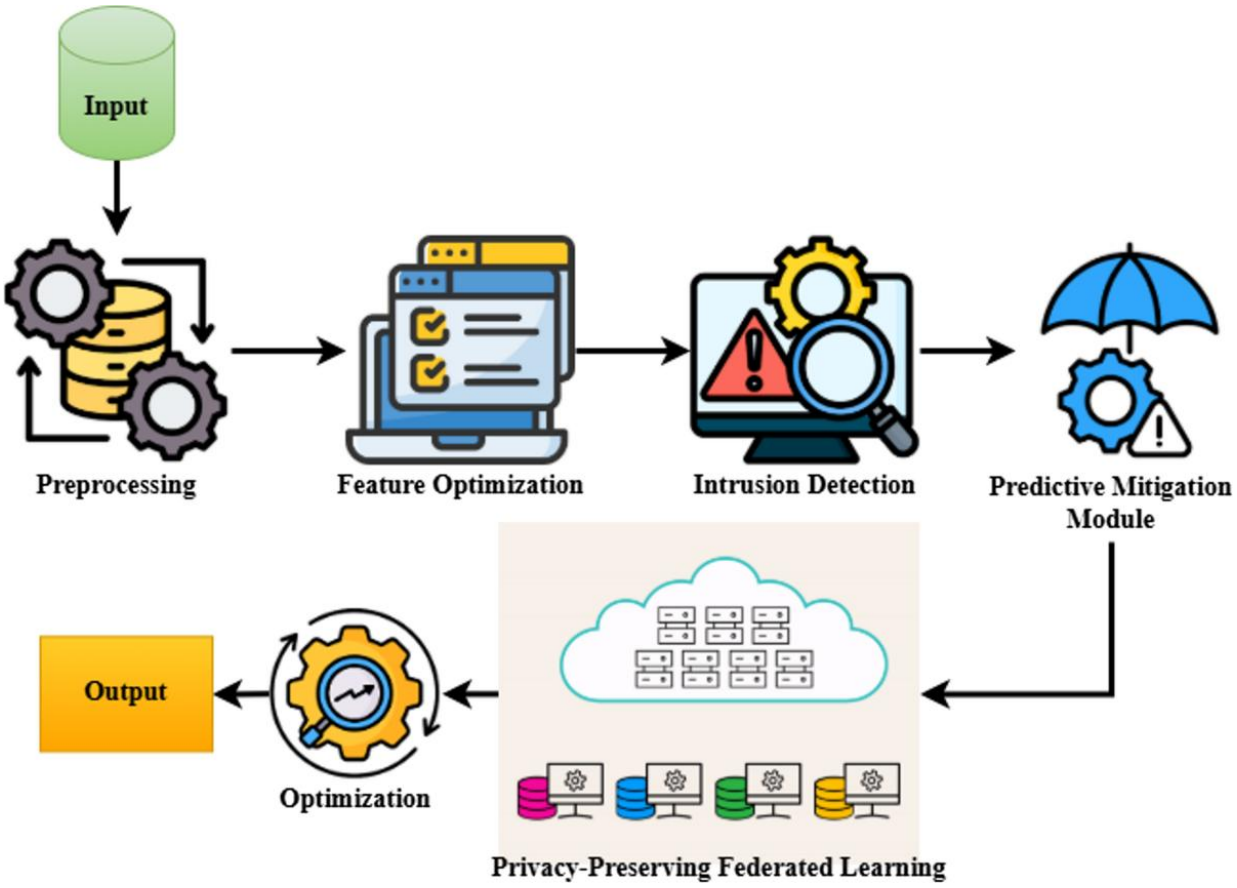

Fig.1.AI-driven intrusion detection and mitigation framework for software-defined IoT networks

The third component integrates explainability methods into the security validation workflow and evaluates their ability to communicate model decisions. SHAP is used to estimate the contribution of individual features to a prediction, while LIME may be applied to generate locally interpretable approximations for selected security findings. Feature importance analysis provides a broader view of variables influencing predictions across the evaluated dataset. These techniques are selected according to model compatibility, computational requirements, and the type of explanation required. For each relevant finding, the framework produces an explanation containing the predicted risk category, contributing security attributes, supporting configuration evidence, the associated policy or vulnerability reference where available, and a recommended remediation action. For example, a cloud storage resource may receive a high-risk classification because public access is enabled, sensitive data are present, and encryption controls are missing. The explanation should distinguish model-derived evidence from deterministic policy violations and avoid presenting statistical feature contributions as proof of causation. Explanations are incorporated into pipeline reports and security dashboards so that developers and security analysts can inspect findings before deciding whether to approve, reject, or

| ISSN: 2347-8446 | www.ijarcst.org | editor@ijarcst.org |A Bimonthly, Peer Reviewed & Scholarly Journal|

||Volume 9, Issue 5, September-October 2026||

DOI:10.15662/IJARCST.2026.0905004

remediate a deployment. Automated responses are classified according to operational impact. Low-risk, reversible actions may be eligible for policy-governed automation, whereas changes involving production access, critical workloads, or potentially disruptive network restrictions require explicit approval. The methodology evaluates explanation quality through fidelity to the underlying model, consistency across comparable observations, agreement with known security evidence, and usefulness for understanding the recommended action. Where human evaluation is feasible, representative developers, security practitioners, and cloud engineers review anonymized explanation examples using a structured assessment protocol. Their feedback is used to assess comprehensibility, perceived relevance, and actionability rather than assuming that a technically detailed explanation is necessarily useful. The study also examines whether explanations remain stable under small, security-irrelevant changes to input records and whether they identify meaningful indicators when the model encounters previously unseen configurations. Potential weaknesses, including misleading feature attribution, model drift, adversarially manipulated inputs, and excessive reliance on generated explanations, are documented as part of the evaluation.

The fourth component focuses on experimental validation, comparative analysis, and assessment of the proposed framework under representative enterprise cloud scenarios. Three principal configurations are compared: a conventional rule-based security validation pipeline, a machine learning-based pipeline without explicit explanations, and an explainable AI-enhanced pipeline combining predictive analysis with interpretable evidence and policy-based controls. Where resources permit, the configurations are evaluated on identical test scenarios, infrastructure definitions, and security labels to support fair comparisons. Quantitative metrics include precision, recall, F1-score, false-positive rate, false-negative rate, and area under the receiver operating characteristic curve when appropriate for the prediction task. Precision measures the proportion of detected findings that are valid, while recall measures the proportion of known security weaknesses successfully identified. These metrics are interpreted alongside operational indicators, including average validation latency, explanation-generation overhead, resource consumption, processing throughput, and the proportion of findings resolved before deployment. Calibration measures may be used to determine whether predicted risk probabilities correspond reasonably to observed outcomes. Explanation quality is evaluated separately using model fidelity, stability, evidence alignment, and structured human assessments. Results are reported across relevant categories, such as cloud configuration weaknesses, identity and access management violations, container vulnerabilities, and dependency risks, to identify where the framework performs effectively and where it remains limited. Statistical comparisons may use confidence intervals and appropriate paired significance tests when the sample size, experimental design, and distributional assumptions support them. Ablation experiments examine the contribution of contextual features, explainability components, and deterministic policy controls to overall performance. Scalability tests evaluate how validation latency changes as the number of resources, security findings, and concurrent pipeline executions increases. The methodology also incorporates a risk-based review of implementation challenges, including integration with existing development tools, cross-cloud interoperability, sensitive security-data exposure, model maintenance, and the possibility of incorrect automated remediation. Reproducibility is supported by documenting dataset characteristics, feature definitions, model configurations, evaluation procedures, and experimental parameters, subject to confidentiality and licensing restrictions. Because empirical results must be obtained through actual experiments, no performance improvements are assumed in advance. The final analysis determines whether explainable AI offers measurable benefits beyond conventional automation and non-explainable machine learning, identifies the operational costs of explanation generation, and establishes the conditions under which the proposed framework may support reliable, transparent, and scalable security validation across enterprise cloud infrastructure platforms.

IV. RESULTS AND DISCUSSION

The integration of Explainable Artificial Intelligence (XAI) into automated DevSecOps security validation provides a structured approach to improving security transparency, vulnerability detection, and compliance verification across enterprise cloud infrastructure platforms. Modern enterprise environments frequently operate across public, private, and hybrid clouds, integrating containerized applications, microservices, infrastructure-as-code (IaC), continuous integration and continuous deployment (CI/CD) pipelines, APIs, and identity management systems. Although DevSecOps integrates security into software development and deployment, traditional validation mechanisms often depend on static rules, predefined vulnerability signatures, and isolated scanning tools. These mechanisms can identify known weaknesses but may provide limited contextual explanations for complex security findings, generate excessive false-positive alerts, or struggle to correlate risks across infrastructure components. The proposed framework combines AI-based security analysis with explainable decision-making and automated validation to address these limitations. It collects information from source code repositories, dependency manifests, container images, cloud configuration files, deployment pipelines, runtime monitoring systems, and compliance policies. Machine learning models analyze the



| ISSN: 2347-8446 | www.ijarcst.org | editor@ijarcst.org |A Bimonthly, Peer Reviewed & Scholarly Journal|

||Volume 9, Issue 5, September-October 2026||

DOI:10.15662/IJARCST.2026.0905004

collected evidence to identify potentially vulnerable code, insecure infrastructure configurations, exposed credentials, excessive permissions, and deviations from established security baselines. Explainability mechanisms subsequently identify the factors contributing to each prediction, enabling developers and security engineers to understand why a particular component has been classified as risky. Techniques such as feature attribution, interpretable rule extraction, and dependency-based risk visualization can connect model predictions with relevant configuration parameters, code fragments, and infrastructure relationships. The automated validation engine then compares identified risks against organizational policies and determines whether a deployment can proceed, requires remediation, or must be escalated for manual review. This approach is expected to improve the interpretability of security decisions while reducing dependence on fragmented manual inspections. However, because no experimental dataset or implementation measurements have been provided, these outcomes represent the anticipated contributions of the proposed framework rather than empirically established improvements. Its actual effectiveness depends on training data quality, the suitability of the explanation methods, the coverage of security scanners, and the consistency of validation policies across cloud providers. Therefore, evaluation should consider detection precision, recall, false-positive rate, explanation fidelity, validation latency, vulnerability remediation time, and policy enforcement consistency. These indicators can establish whether explainable AI provides practical advantages over conventional DevSecOps security validation without introducing unacceptable computational overhead or deployment delays.

The second major contribution of the framework is the use of explainable risk assessment to prioritize security vulnerabilities according to their technical characteristics, infrastructure context, and potential business impact. Enterprise cloud environments contain numerous interconnected resources, including virtual machines, containers, databases, storage services, network configurations, identity roles, and serverless applications. A vulnerability that appears relatively minor in isolation may become critical when it affects an internet-facing service, a privileged workload, or a component connected to sensitive business information. Conversely, a vulnerability identified by a scanner may have limited practical exposure when appropriate compensating controls are in place. Conventional severity scores provide useful initial guidance, but they may not adequately represent the relationships among vulnerabilities, deployment configurations, accessibility, and business criticality. The proposed framework addresses this issue by combining vulnerability findings with dependency graphs, asset inventories, access-control information, network exposure, and organizational risk policies. AI models can estimate risk using these contextual features, while explainability techniques identify the evidence that contributes to the resulting classification. For example, an insecure container configuration may receive a higher risk assessment when the container runs with elevated privileges, exposes a sensitive service, and has access to confidential data. The explanation can identify these contributing conditions and recommend specific corrective actions, such as removing unnecessary privileges, restricting network access, updating vulnerable dependencies, or modifying infrastructure policies. In infrastructure-as-code validation, explainable analysis can identify which configuration attributes violate security requirements and demonstrate how the violation affects deployment risk. During CI/CD execution, automated gates can block deployments containing explicitly prohibited weaknesses, while lower-risk findings can be recorded for scheduled remediation. This risk-based approach may improve prioritization by directing engineering effort toward vulnerabilities with the greatest potential impact. Nevertheless, explainability must not be confused with correctness. Feature-attribution methods may produce unstable explanations, and a plausible explanation does not prove that a model's prediction is accurate or causally justified. Risk scores can also be distorted by incomplete asset inventories, outdated vulnerability databases, or inaccurate business-impact classifications. To address these issues, the framework should combine AI-generated assessments with deterministic security rules, authoritative vulnerability intelligence, documented asset ownership, and expert review for critical decisions. Explanation quality should be assessed using fidelity, stability, completeness, and usefulness to security practitioners. In addition, every validation result should retain the underlying evidence, model version, relevant policy, and recommended remediation so that developers can reproduce and audit the decision. Such measures can improve transparency and support more consistent security governance across heterogeneous cloud infrastructure platforms.

Automated security validation and cross-platform DevSecOps integration constitute the operational foundation of the proposed framework. In contemporary software delivery environments, security checks must be executed repeatedly throughout the development lifecycle without creating unnecessary deployment bottlenecks. The framework integrates explainable AI into development, build, testing, staging, and production deployment stages, enabling security findings to be identified and addressed as early as possible. During source-code analysis, the system can inspect potentially vulnerable programming patterns, insecure API implementations, and accidental exposure of sensitive credentials. During dependency analysis, it can identify known vulnerabilities in external libraries and explain their potential relevance to the application. Container validation can assess image provenance, outdated packages, unnecessary

| ISSN: 2347-8446 | www.ijarcst.org | editor@ijarcst.org |A Bimonthly, Peer Reviewed & Scholarly Journal|

||Volume 9, Issue 5, September-October 2026||

DOI:10.15662/IJARCST.2026.0905004

privileges, and configuration weaknesses, while infrastructure-as-code scanning can identify insecure network rules, public storage exposure, and excessive permissions. Before deployment, policy enforcement mechanisms evaluate whether the identified issues satisfy predefined acceptance criteria. High-risk violations can trigger a deployment block, whereas lower-risk findings can generate remediation tickets or require explicit approval. After deployment, runtime telemetry can be incorporated into subsequent assessments to identify configuration drift, suspicious behavior, and emerging risks that were not visible during development. Cross-platform integration requires normalized security findings, consistent identity and access controls, standardized interfaces, and platform-specific connectors for different cloud providers and DevSecOps tools. Explainable outputs should remain traceable as applications move between development, testing, and production environments. The system should also support versioned security policies, reproducible validation results, and auditable exception handling. However, extensive AI analysis can increase pipeline execution time, and overly restrictive automated gates may delay legitimate releases. Similarly, inconsistent security controls across cloud platforms can produce conflicting assessments if equivalent resources are represented differently. Performance should therefore be evaluated through pipeline latency, validation throughput, policy coverage, successful remediation rates, deployment failure rates attributable to security checks, and cross-platform consistency. Additional measurements should examine whether explanations help developers resolve vulnerabilities more quickly and whether the automated framework reduces repeated findings across development stages. Human oversight remains essential for ambiguous findings, business-critical deployments, and exceptions involving accepted residual risk. Overall, explainable AI offers a promising means of connecting automated vulnerability analysis with understandable security decisions, supporting more transparent and repeatable DevSecOps practices across enterprise cloud infrastructure. Its practical value must ultimately be established through controlled experiments, realistic workloads, and comparisons with established scanning and policy-enforcement approaches.

V. CONCLUSION

Explainable AI for Automated DevSecOps Security Validation Across Enterprise Cloud Infrastructure Platforms presents an integrated approach to strengthening cloud security by combining intelligent vulnerability assessment, transparent risk interpretation, and automated policy enforcement throughout the software delivery lifecycle. As enterprises increasingly adopt multicloud architectures, containerized workloads, microservices, and automated deployment pipelines, the number and complexity of security validation requirements continue to grow. Traditional security tools remain important for detecting known vulnerabilities and enforcing predefined controls, but their findings may be fragmented across development and infrastructure teams. In addition, some AI-based security solutions provide predictions without sufficient explanation, making it difficult for developers and security administrators to determine why a particular issue was identified, how it affects the business, or which corrective action should be prioritized. The proposed framework addresses these challenges by incorporating explainability into automated security validation. It collects security evidence from source code, software dependencies, container images, infrastructure configurations, identity systems, and cloud runtime environments. AI models use this evidence to identify potentially risky conditions, while explanation mechanisms connect predictions to relevant technical attributes, configuration weaknesses, and infrastructure relationships. Automated validation then evaluates the findings against predefined security requirements and determines whether a deployment should proceed, be restricted, or undergo further review. This combination can improve the traceability of security decisions and support more consistent risk management across development and operations teams. A key advantage of the approach is its potential to transform security validation from a collection of isolated scanning activities into a coordinated, evidence-based process. Developers can receive actionable explanations during code review, infrastructure engineers can understand configuration violations before deployment, and security teams can prioritize findings according to technical severity and business context. Nevertheless, explainable AI should be considered a complementary capability rather than a substitute for established security controls. An explanation may be incomplete, misleading, or inconsistent with the actual behavior of a model. Consequently, reliable implementation requires validated detection methods, authoritative security rules, high-quality asset inventories, and clearly defined risk acceptance criteria. Since no empirical measurements have been supplied, the framework's expected benefits remain hypotheses requiring systematic testing. Future experiments should compare it with conventional static analysis, vulnerability scanning, and rule-based DevSecOps validation using standardized workloads and repeatable evaluation procedures. Such investigations would establish whether explainability improves detection quality, remediation efficiency, developer understanding, and deployment reliability in practical enterprise environments.

The broader contribution of the framework is its emphasis on making automated security decisions understandable, auditable, and operationally actionable. Security automation becomes more effective when developers and administrators can understand the evidence behind a finding rather than merely receiving a severity score or an

| ISSN: 2347-8446 | www.ijarcst.org | editor@ijarcst.org |A Bimonthly, Peer Reviewed & Scholarly Journal|

||Volume 9, Issue 5, September-October 2026||

DOI:10.15662/IJARCST.2026.0905004

unexplained deployment failure. Explainable outputs can support this objective by identifying contributing features, linking findings to relevant configuration parameters, and providing remediation recommendations that can be verified against established security policies. Dependency-aware risk analysis can further reveal how a weakness in one resource affects connected applications, privileged services, or sensitive datasets. These capabilities are particularly relevant in multicloud environments, where security teams must interpret findings produced by different infrastructure platforms and ensure consistent enforcement across heterogeneous services. However, achieving consistency requires normalized data models, standardized reporting formats, and platform-specific adapters that account for differences in cloud services and configuration semantics. The framework should also preserve a clear separation between AI recommendations and authoritative security decisions. Deterministic rules should continue to enforce mandatory requirements, while AI-based assessments can provide additional context and prioritization. High-impact actions, including blocking critical production releases, modifying privileged access, or isolating business-critical workloads, should follow documented approval procedures and recovery policies. In addition, organizations must protect the confidentiality and integrity of security telemetry, source-code evidence, vulnerability reports, and model artifacts. Least-privilege access, encryption, audit logging, secure model management, and appropriate data-retention policies are essential to prevent the validation framework from becoming a new attack surface. The quality of the system should be assessed not only through conventional detection metrics but also through explanation fidelity, stability, reproducibility, and usefulness to practitioners. Continuous evaluation is necessary because software dependencies, cloud configurations, attack techniques, and organizational policies change over time. Model updates and security policy changes should therefore be versioned, tested, and monitored before broad deployment. From an operational perspective, organizations can introduce the framework incrementally by beginning with explainable reporting and advisory recommendations, progressing to automated low-risk remediation, and enabling stricter deployment gates after sufficient validation. This approach reduces the likelihood of disrupting legitimate business operations while building confidence in the system. Ultimately, explainable AI can help bridge the gap between intelligent threat assessment and trustworthy security automation. Its success depends on combining transparent reasoning with reliable detection, responsible governance, cross-platform interoperability, and measurable operational benefits.

A further conclusion is that explainability should be treated as a measurable security requirement rather than a presentation feature added after model development. An explanation is useful only when it accurately reflects the evidence supporting a prediction, remains sufficiently stable under reasonable changes in input, and enables practitioners to take appropriate action. In security-sensitive environments, misleading explanations can create false confidence and encourage developers to overlook important weaknesses. The proposed framework should therefore evaluate explanation quality alongside vulnerability detection performance and operational efficiency. Feature attribution, rule extraction, dependency visualization, and counterfactual analysis may each be useful for different validation tasks, but no single technique is guaranteed to provide adequate explanations for every model or cloud configuration. Selection should depend on the model architecture, the type of security evidence, and the decision being supported. Counterfactual explanations, for example, may help show which configuration changes could reduce a risk classification, but proposed changes must still be checked against actual security requirements and application functionality. Similarly, natural-language remediation guidance should be linked to verified evidence and established documentation rather than generated without validation. The framework should also support disagreement between AI assessments and deterministic scanners by preserving the original findings and directing conflicting results to additional review. This is important because different tools may assess different aspects of the same resource, and an apparently contradictory result may reflect incomplete context rather than a simple model error. Furthermore, successful deployment requires organizational collaboration among developers, security engineers, infrastructure administrators, compliance officers, and risk owners. Transparent evidence and consistent reporting can improve communication between these groups, but training and clearly defined responsibilities remain necessary. A mature implementation should document who can approve exceptions, which violations automatically block releases, how model errors are reported, and how security decisions are audited. These measures support accountability and reduce the risk of uncontrolled automation. In conclusion, the proposed framework offers a foundation for developing more transparent, context-aware, and reliable DevSecOps security validation across enterprise cloud platforms. By combining explainable intelligence with deterministic controls and automated workflows, it can support informed remediation and consistent security governance. Demonstrating its real-world value will require reproducible experiments, rigorous explanation assessment, and long-term evaluation of its impact on vulnerability management, deployment performance, and enterprise security outcomes.

| ISSN: 2347-8446 | www.ijarcst.org | editor@ijarcst.org |A Bimonthly, Peer Reviewed & Scholarly Journal|

||Volume 9, Issue 5, September-October 2026||

DOI:10.15662/IJARCST.2026.0905004

VI. FUTURE WORK

Future research should prioritize the development of a working prototype and the quantitative evaluation of explainable AI within realistic DevSecOps pipelines. The prototype could integrate source-code analysis, software composition analysis, container scanning, infrastructure-as-code validation, cloud configuration assessment, and CI/CD orchestration. Experiments should use representative vulnerabilities, insecure deployment configurations, vulnerable dependencies, and misconfigured identity permissions across multiple cloud platforms. Detection performance should be evaluated using precision, recall, F1-score, false-positive rate, and vulnerability coverage, while operational performance should include pipeline latency, remediation time, validation throughput, and deployment interruption rates. Explainability should be assessed using fidelity, stability, completeness, reproducibility, and developer usefulness. Comparative experiments should examine conventional security scanners, AI-based detection without explanations, and the complete explainable validation framework. Researchers should also investigate how explanations influence developer decisions, remediation accuracy, and the prioritization of high-risk vulnerabilities. Adversarial testing should evaluate model evasion, manipulated configuration inputs, poisoned training datasets, and attempts to generate misleading explanations. Testing across different application sizes, deployment frequencies, and infrastructure configurations would help determine scalability and generalizability. Repeated trials and statistical analysis would establish whether the framework provides measurable advantages without introducing excessive computational overhead or delaying software delivery.

Further work should explore adaptive learning, standardized cross-cloud security representations, and trustworthy automated remediation. Explainable graph-based models could improve understanding of relationships among vulnerabilities, infrastructure resources, and business-critical services. Continuous learning and model-drift detection could help maintain detection quality as software dependencies and cloud configurations evolve, while privacy-preserving learning techniques could support collaboration without unnecessarily exposing proprietary source code or sensitive security telemetry. Future systems could incorporate policy-as-code, software supply-chain provenance, and runtime feedback to connect development-time validation with operational security monitoring. Automated remediation could generate proposed code or configuration changes, validate them in isolated environments, and submit them for approval before production deployment. Formal verification and rollback mechanisms should be investigated for high-impact changes. Research should also establish common benchmarks for evaluating explanation quality, security effectiveness, interoperability, and computational cost across heterogeneous cloud platforms. Finally, long-term deployment studies should examine regulatory compliance, maintainability, developer adoption, and total cost of ownership. These advances could help establish explainable AI as a dependable component of enterprise DevSecOps, improving the transparency and reliability of automated security validation while preserving human accountability and operational resilience.

REFERENCES

1) Shamshirband, S., Fathi, M., Chronopoulos, A. T., Montieri, A., Palumbo, F., & Pescapè, A. (2020). Computational intelligence intrusion detection techniques in mobile cloud computing environments: Review, taxonomy, and open research issues. Journal of Information Security and Applications, 55, Article 102582. https://doi.org/10.1016/j.jisa.2020.102582
2) Sazu, A. H., Khan, M. A. N., Sultana, S., Tasnim, M., Jakir, T., Alam, A., ... & Himeluzzaman, M. (2025). Artificial intelligence-based anomaly detection for environmental and financial risk management in banking. International Journal of Advances in Signal and Image Sciences, 2083–2095.
3) Singh, V. (2026). Enterprise Resource Planning for Financial Applications: Engineering, Architecture, and Design Principles. International Journal of Scientific Research in Science and Technology, 13(2), 653-661.
4) Venkatasalam, K., Rajendran, P., & Thangavel, M. (2019). Improving the accuracy of feature selection in big data mining using accelerated flower pollination (AFP) algorithm. Journal of medical systems, 43(4), 96.
5) Neela, S. (2025). A next-generation middleware architecture for seamless enterprise integration and real-time analytics. International Journal of Engineering & Extended Technologies Research, 7(6), 1332–1336.
6) Kollu, R. K. (2023). CRM architecture deep dive: Building for scale on Salesforce. International Journal of Engineering & Extended Technologies Research, 5(5), 7309–7312.
7) Yasin, M., Akter, S., Khan, M. A. N., Tasnim, M., Sultana, S., Jakir, T., ... & Akter, K. S. (2026). XAI-Enabled Intelligent Detection and Mitigation of Zero-Day Attacks in Healthcare IoT. International Journal of Future Innovative Science and Technology (IJFIST), 9(5), 1807.

| ISSN: 2347-8446 | www.ijarcst.org | editor@ijarcst.org |A Bimonthly, Peer Reviewed & Scholarly Journal|

||Volume 9, Issue 5, September-October 2026||

DOI:10.15662/IJARCST.2026.0905004

8) Konatham, M. R., Uddandarao, D. P., Vadlamani, R. K., & Konatham, S. K. R. (2025, July). Federated Learning for Credit Risk Assessment in Distributed Financial Systems using BayesShield with Homomorphic Encryption. In 2025 International Conference on Computing Technologies & Data Communication (ICCTDC) (pp. 1-6). IEEE.
9) Selvarajan, K. (2025). Petabyte-scale data processing for real-time enterprise security analytics. International Research Journal of Innovative Engineering (IRJIE), 9(5), 17566–17576.
10) Chinthamani, R. (2026). Intelligent observability and predictive failure detection for large-scale cloud-native enterprise systems. International Journal of Research Publications in Engineering, Technology and Management, 9(1), 268–272.
11) Pothuri, M. K. (2026). AI-Optimized Symmetry Episode Analytics for Early Detection of High-Utilizers: A Claims-Based Predictive Modelling Framework Using Advanced Machine Learning Models. International Journal of Artificial Intelligence, Data Science, and Machine Learning, 7(1), 279-287.
12) Patel, K., Boda, S. S., Gunasekaran, R. M. N., Alluri, V. R. R., & Kondru, V. S. A. (2026, April). Genetic Algorithm–Optimized Deep Learning for Intelligent Cloud–Fog Architectures. In 2026 IEEE 15th International Conference on Communication Systems and Network Technologies (CSNT) (pp. 345-351). IEEE.
13) Mathew, A. (2026). AI-Driven Incident Response Using Collaborative Cyber Security Analytics for Enterprise Cloud Environments. International Journal of Research Publications in Engineering, Technology and Management (IJRPETM), 9(2), 676-683.
14) Madabathula, L. (2025). A governance-driven enterprise lakehouse architecture for real-time analytics and machine learning enablement. International Journal of Applied Mathematics, 38(12s), 5022–5033.
15) Karrothu, A. (2024). A cloud-native architecture for scalable real-time security telemetry ingestion and threat signal processing. Journal of Computational Analysis and Applications (JoCAAA), 33(8), 8137-8154.
16) Ahuja, D. (2026, April). Declarative Multi-Cluster Kubernetes Deployment Architecture Using GitOps. In 2026 International Symposium of Systems, Advanced Technologies and Knowledge (ISSATK) (pp. 1-6). IEEE.
17) Bandari, U., & Gajendran, R. (2024). From data hub to data marketplace: Retrieval-augmented generation over governed KPI products in a federated customer experience platform. International Journal of Research Publications in Engineering, Technology and Management, 7(3), 10616–10626.
18) Aragani, V. M., Maroju, P. K., & Mudunuri, L. N. R. (2024, December). Implementing Non-functional Production Regression Testing with Kubernetes. In International Conference on Information and Communication Technology for Competitive Strategies (pp. 231-244). Singapore: Springer Nature Singapore.
19) Sharma, K., Singhal, A. B., karthik Chundi, V. R., & Arveti, S. (2026, February). Analyzing Interdependencies Between IoT Data Integration Capabilities and Real-Time Operational Decision-Making: A DEMATEL Approach. In 2026 5th International Conference on Innovative Practices in Technology and Management (ICIPTM) (pp. 1-5). IEEE.
20) Sahu, S. K. (2024). AI-Driven Governance and Privacy-Preserving Architecture for Healthcare CRM in Multi-Org Cloud Environments: A Human-Centered Big Data Framework for Trustworthy Decision-Making. International Journal of Artificial Intelligence & Digital Transformation, 7(2), 01-06.
21) Bitragunta, S. L. V. (2025). Optimizing hybrid energy storage in EVs using AI enhanced fuzzy logic control systems. International Journal of Science, Research and Technology, 8(1), 13525-13533.
22) Chinnam, N. B. (2022). Engineering real-time supply chain visibility and ETA intelligence platforms for large-scale retail distribution networks. International Journal of Research Publications in Engineering, Technology and Management (IJRPETM), 5(6), 7918–7922.
23) Sugumar, R. (2026). Adaptive Recommendation Systems Using Machine Learning for Dynamic User Preference Modeling in Online Environments. International Journal of Research and Applied Innovations, 9(2), 167-176.
24) Gadige, C. D. (2024). Event-driven CRM architecture for real-time order fulfillment and product provisioning. International Journal of Engineering & Extended Technologies Research, 6(2), 7968–7978.
25) Ramasamy, M. (2022). Closed-loop network automation: Architectural principles for intent-based networking. International Journal of Future Innovative Science and Technology (IJFIST), 5(5), 9284–9294. https://doi.org/10.15662/IJFIST.2022.0505010
26) Kundavaram, R. R., Bandhela, R. R., & Onteddu, A. R. (2025). Quantum support vector regression for high-dimensional data: A hybrid quantum-classical approach (SSRN Scholarly Paper No. 5414059). SSRN. https://doi.org/10.2139/ssrn.5414059
27) Rengarajan, A. (2026). Cloud-Native Intelligent Quality Assurance and Predictive Analytics for API-Centric Financial Systems Using Machine Learning. International Journal of Humanities and Information Technology, 8(2), 30-37.
28) Gudipuri, N. (2024). Intent-Aware Enterprise Architecture Using Persistent AI Agents for Dynamic FinTech Service Composition. The Eastasouth Journal of Information System and Computer Science, 2(01), 151-161.



| ISSN: 2347-8446 | www.ijarcst.org | editor@ijarcst.org |A Bimonthly, Peer Reviewed & Scholarly Journal|

||Volume 9, Issue 5, September-October 2026||

DOI:10.15662/IJARCST.2026.0905004

29) Balla, R. T., Padmanabham, S., Bhandari, R. S., Kumar, V., & Khattar, S. (2026, February). Scalable IoT Frameworks for Smart Grid Management: A Time-Series and Predictive Maintenance Analysis Using ARIMA Models. In 2026 5th International Conference on Innovative Practices in Technology and Management (ICIPTM) (pp. 1-6). IEEE.
30) Chandran, A. S. (2023). A unified Spark ETL stack across Cloudera CDP, Azure HDInsight, and Databricks. International Journal of Future Innovative Science and Technology, 6(6), 11769–11779.
31) Ravichandran, S., & Kandasamy, V. (2025). Optimized Attention Augmented Residual Convolutional Neural Network with Fa-Resnet for Fabric Defect Detection. Journal of Control Engineering and Applied Informatics, 27(4), 3-15.
32) Shrestha, G., Ivanov, T., Vemireddy, K., Pyayt, A., & Gubanov, M. (2026, July). Hemolix. TabGen: Optimized Table Generation from Documents. In Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (ACL 2026) (pp. 1055-1066).
33) Nelavala, M. M. (2026). Blockchain-Enhanced Event Sourcing for Immutable Audit Trails in Core Banking Systems: A Merkle-Anchored Hybrid Architecture. Journal of Intelligent Decision Making and Information Science, 3(10s), 638-661.
34) Karuturi, Y. (2025). Serverless computing framework for enterprise ERP background job processing in cloud SAP environments. International Journal of Science, Research and Technology, 8(2), 13938–13947.
35) Puram, S. (2024). AI-Enabled Software-Defined Vehicles: An Intelligent Cloud-Edge Architecture for Adaptive, Secure, and Resilient Automotive Software Systems. The USA Journals TAJAS, 6(03), 23-33.
36) Mudunuri, L. N. R., Aragani, V. M., & Maroju, P. K. (2024, December). Development of an AI-Powered Garbage Detection System for Environmental Sustainability Via YOLOv5. In International Conference on Information and Communication Technology for Competitive Strategies (pp. 317-327). Singapore: Springer Nature Singapore.
37) Kalabhavi, V. (2023). Enterprise quote management using AI for product configuration, pricing and approval workflows in SAP CPQ. International Journal of Science, Research and Technology, 6(1), 9231–9242.
38) Reddy, K. V. (2024). Optimizing gate-level simulation performance through cloud-based distributed computing. International Journal of Research Publications in Engineering, Technology and Management, 7(6), 11747–11752.
39) Kasarapu, B. C. (2026). Generative AI-Enabled Micro-Frontend Framework for Scalable and Intelligent Enterprise Retail Applications. International Journal of Computer Information Systems and Industrial Management Applications, 18(5s), 297-309.
40) Shaik, N. (2025). Securing Kubernetes: An integrated approach to AI-driven threat detection and eBPF-based security monitoring. International Journal of Research and Applied Innovations, 8(2), 11289–11294.