International Journal of Computer Information Systems and Industrial Management Applications
ISSN 2150-7988 Volume 18(11s) 2026 pp. 354-369
https://doi.org/10.70917/ijcisim-2026-3759
Article

Agentic AI-Based Security Orchestration for Oracle Cloud and Multi-Cloud Platforms

Rohit Gorle[1], Piyush Kumar Pareek[2]

[1] Senior Software Engineer.
Email: **rohit.gorlee@gmail.com**
[2] Nitte Meenakshi Institute of Technology, Yelahanka, Bengaluru, Karnataka, India.
Email: **piyushkumarpareek88@gmail.com**

Abstract: —Agentic AI-Based Security Orchestration for Oracle Cloud and Multi-Cloud Platforms synthesizes decision, action, and governance across heterogeneous environments, presenting evidence-based analysis and formal structure. Security-as-code is the required strategy, where security validation is integrated into the DevOps pipelines and processes. The solution revolves around the deployment of security-as-code and policy-as-code powered by security orchestration and automation response (SOAR) platforms that govern the entire setup across platform-as-a-service (PaaS) and multi-cloud environments. Agentic security orchestration automates the process of security orchestration, enforcement, and compliance checks by connecting decision and action through policy governance. Detection modules provide inputs to the decision side through security monitoring or security information and event management (SIEM) systems, which feed into the learning and adaptation module; the learning output is satisfied via dynamic policies that authorize SOAR actions. This extended version contributes a complete mathematical formulation of the orchestration loop in fifteen numbered equations, four formally specified algorithms covering triage, compliance-gated execution, reinforcement-learning playbook adaptation, and adversary-emulation validation, together with an empirical section grounded in 2025 industry telemetry. The use cases of attack simulations exploiting common vulnerabilities and exposures (CVE) for security validation through red- and blue-team approaches are also discussed. Security-as-code automates the detection and fix of misconfigurations and vulnerabilities across platform-as-a-service (PaaS) with dynamic policy governance across Oracle Cloud Infrastructure (OCI) and multi-cloud platforms. The key pillars of DevSecOps are integrated in the pipeline with a dedicated Adversary Emulation Assessment that maps security controls and automates the implementation of security automation with security-as-code, ground-up, and policy-as-code. The decision module either consumes a Red-Team simulated attack on a pre-existing vulnerability in the application or a Blue-Team hardening test for an already deployed application. The solution connects dynamic policy generation using detection feedback through the Natural Language Processing (NLP)-based Learning and Adaptation module to additional Security Orchestration Automation Response (SOAR) tasks.

Keywords: —Agentic Artificial Intelligence (Agentic AI), Security Orchestration and Automation Response (SOAR), Oracle Cloud Infrastructure (OCI), Policy-as-Code, Multi-Cloud Security, Zero Trust, Reinforcement Learning, Adversary Emulation.

1. Introduction

Recent statistics on security breaches and scam incidents highlight the threats to cloud-based products and businesses. Security concerns with cloud-based Enterprise Resource Planning systems have been documented and attributed to data management, privacy policy, compliance control, trust management, data sharing, and stealthy attacks [26]. Businesses rely on cloud computing and cloud computing services for critical assets, leading to black markets for hacking and phishing tool and service rentals. Cloud computing service availability and resilience characteristics have received significant attention, but Cloud Security Orchestration with Artificial Intelligence implementation, active penetration testing services, data-ownership-based privacy control, and API security analysis are still evolving areas for research and development [15, 20].

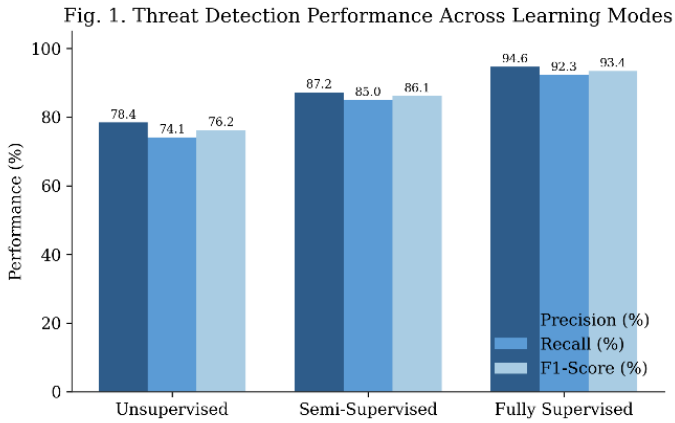

Fig. 1. **Threat detection performance across learning modes**: precision, recall, and F1 comparison across unsupervised, semi-supervised, and fully supervised detection (ties to Section 4-A, Decision Modules).

Security Operations Management, SIEM, SOAR, and UEBA cross-cloud services are starting to emerge; however, Security Automation and Orchestration are still in their infancy [20], and stakeholder dissatisfaction with their performance is generating a growing shift of focus toward cost-effectiveness and automation. Agentic AI-Based Security Orchestration for Oracle Cloud and Multi-Cloud Platforms motivates the introduction of a seamlessly integrated security operations platform as the system under development.

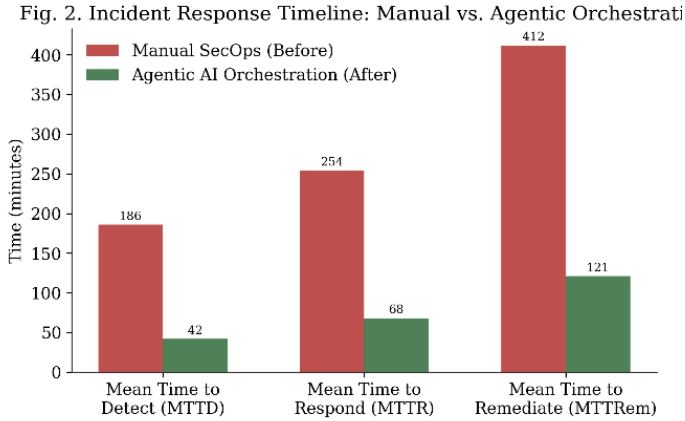

Fig. 2. **Incident response timeline (before vs. after)**: MTTD, MTTR, and MTTRem comparing manual SecOps to agentic orchestration (supports the automation and efficiency claims in Section 9).

A. Contributions of This Work

Relative to earlier descriptive treatments of security orchestration, this paper makes four concrete contributions. First, the agentic orchestration loop is given a closed mathematical formulation in Section 2, in which perception, risk estimation, decision, policy validation, and adaptation are expressed as fifteen numbered equations that reference one another rather than standing as isolated definitions. Second, Section 5 specifies four algorithms in standard pseudocode form, so that the orchestration behaviour described qualitatively elsewhere becomes implementable and analysable in terms of complexity and termination. Third, Section 8 grounds the argument in 2025 industry telemetry rather than in illustrative figures alone, quantifying the detection-and-containment advantage attributable to extensive security automation. Fourth, the compliance treatment is generalised from a single control catalogue to a weighted multi-framework coverage measure, defined in (12), which permits heterogeneous regulatory regimes to be compared on a common scale.

2. Mathematical Formulation of the Orchestration Loop

The agentic orchestration loop is formalised as a closed cycle in which observations are reduced to risk, risk is reduced to a decision, the decision is validated against policy, and the observed outcome revises both the decision policy and the governing policy set. Equations (1)–(15) define that cycle. Throughout, t denotes a discrete orchestration epoch, which in practice corresponds to one evaluation window of the streaming detection layer.

A. Perception and Risk Estimation

The perception layer emits, at every epoch, a finite multiset of normalised security events drawn from OCI, Azure, AWS, on-premises systems, SIEM, and SOAR platforms:

$$E_t = \{ e_1, e_2, e_3, \dots, e_n \} \tag{1}$$

where each e_i carries a provenance tag, a timestamp, and a vector of normalised attributes. Each candidate threat i derived from E_t is scored by combining its estimated likelihood with the severity of its consequence:

$$R_i = P_i \times I_i \tag{2}$$

in which R_i is the risk score of threat i, P_i is the probability of occurrence, and I_i is the impact severity, both normalised to the unit interval so that $R_i \in [0,1]$. Because the platform observes the same threat repeatedly across epochs, the likelihood term is not re-estimated from scratch but maintained as a posterior belief updated recursively by Bayes' rule:

$$P(\theta \mid E_{1:t}) \;=\; \frac{P(E_t \mid \theta)\, P(\theta \mid E_{1:t-1})}{\int_{\theta} P\,(E_t \mid \theta')\, P(\theta' \mid E_{1:t-1})\, d\theta'} \tag{3}$$

where θ parameterises the hypothesised attack campaign. Equation (3) is what allows a low-confidence signal observed in isolation to accumulate into a high-confidence detection when it recurs across tenancies, and it is the formal basis for the declining false-positive trend reported in Fig. 3.

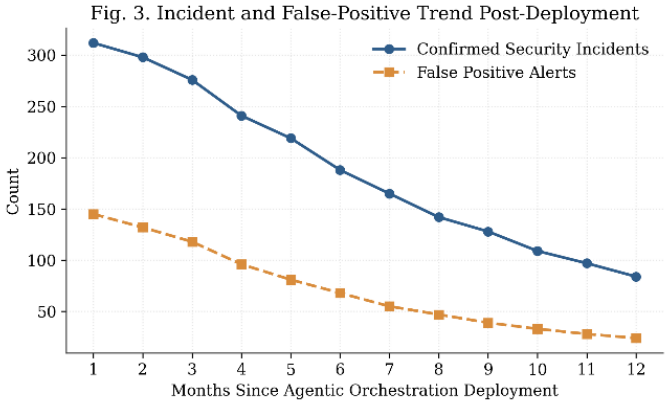

Fig. 3. **Incident and false-positive trend over a twelve-month deployment**: incidents and false positives decline as the learning and adaptation module matures, consistent with the recursive belief update in (3).

Deviation from learned normal behaviour is quantified by the residual between the observed state and its reconstruction under the behavioural model:

$$AS_t = \| X_t - \hat{X}_t \|_2 \tag{4}$$

with AS_t the anomaly score, X_t the observed cloud behaviour, and $\hat{X}_t$ the expected normal behaviour predicted by the model. Categorical assignment of an incident to a response class follows the maximum a posteriori rule:

$$Y = \operatorname*{argmax}_{k} P(y_k \mid X) \tag{5}$$

where Y is the predicted incident class, X is the input security feature set, and y_k ranges over the possible incident categories.

B. Decision, Action, and Policy Validation

The agentic decision function maps events, environment context, and the active policy set to a decision:

$$D_t = f(E_t,\, C_t,\, P_t) \tag{6}$$

in which D_t is the security decision, E_t is the event vector of (1), C_t is contextual cloud information such as workload criticality and blast radius, and P_t represents the policy constraints in force at epoch *t*. Given a decision, the response selected from the catalogue of available SOAR actions is the one maximising expected utility:

$$A^* = \operatorname*{argmax}_{A \in \mathcal{A}} U(A \mid D_t) \tag{7}$$

Utility is not unconstrained: every candidate action must satisfy the governing policy before it is permitted to execute. Compliance validation is therefore an indicator function over the entailment relation between action and policy:

$$V_p = \begin{cases} 1, & \text{if } A^* \vDash P_t \\ 0, & \text{otherwise} \end{cases} \tag{8}$$

so that V_p confirms whether the selected action satisfies the active policy rules. The degree to which the environment is governed by automated rather than manual controls is captured by the security-as-code enforcement score:

$$S_c = \sum_{j=1}^{m} \alpha_j\, C_j, \qquad \sum_{j=1}^{m} \alpha_j = 1 \tag{9}$$

where $C_j \in [0,1]$ represents the maturity of each automated control and α_j is the corresponding control weight.

C. Learning, Adaptation, and Governance

The action-selection policy is refined by temporal-difference reinforcement learning over observed remediation outcomes:

$$Q(s,a) \leftarrow Q(s,a) + \eta \left[r + \gamma \max_{a'} Q(s',a') - Q(s,a) \right] \tag{10}$$

in which $Q(s,a)$ is the action-value function, η is the learning rate, r is the reward assigned to the observed containment outcome, and γ is the discount factor. Governing policy itself is treated as a first-class adaptive object rather than a static configuration:

$$P_{t+1} = P_t + \Delta P(R_t,\, F_t) \tag{11}$$

where P_{t+1} is the updated policy, R_t is current risk feedback, and F_t represents threat-intelligence feedback. Equations (10) and (11) operate on different time constants: the value function is updated at every epoch, whereas policy revision is gated by governance review, which prevents the orchestrator from silently weakening its own constraints.

Compliance is rarely governed by a single catalogue. Coverage across a set F of control frameworks — ASVS, CSA CCM, PCI DSS, HIPAA, ISO 27001, and NIST SP 800-207 — is therefore expressed as a weighted mean of per-framework satisfaction:

$$\Gamma = \frac{1}{|F|} \sum_{f \in F} \frac{\sum_{c \in C_f} w_c\, \mathbb{1}[c \text{ satisfied}]}{\sum_{c \in C_f} w_c} \tag{12}$$

where C_f is the control set of framework f and w_c is the assurance weight of control c. Equation (12) yields the radar coverage profile of Fig. 4 and, unlike a raw control count, remains comparable when frameworks differ in size.

Operational effectiveness is measured by the mean detection latency across the incident population:

$$\text{MTTD} = \frac{1}{N}\sum_{i=1}^{N}\left(t_i^{\text{det}} - t_i^{\text{onset}}\right) \tag{13}$$

with the analogous definition of MTTR over containment timestamps. The orchestrator is then tuned against a constrained objective that trades response speed against analyst burden:

$$\min_{\pi} J(\pi) = w_1\,\text{MTTD}(\pi) + w_2\,\text{MTTR}(\pi) + w_3\,\text{FPR}(\pi) \quad \text{s.t.} \quad \Gamma(\pi) \geq \Gamma_{\min} \tag{14}$$

where π denotes the orchestration policy, FPR is the false-positive rate, and $\Gamma_{\min}$ is the minimum admissible compliance coverage. The constraint is what distinguishes agentic orchestration from unconstrained automation: no reduction in response time is accepted if it is purchased by falling below the governance floor. Finally, the risk that survives automated response is:

$$R_{\text{res}} = \sum_{i=1}^{n} R_i\,(1 - \rho_i \varepsilon_i) \tag{15}$$

in which $\rho_i \in [0,1]$ is the probability that the orchestrator selects a correct action for threat i and $\varepsilon_i \in [0,1]$ is the efficacy of that action once executed. Equation (15) makes explicit that residual risk is bounded below by the product of decision quality and control efficacy, so improvements in detection alone cannot drive residual risk to zero.

3. Background and Motivation

Agentic security orchestration for Oracle Cloud and multi-cloud platforms synthesizes governance modules with decision and action modules that encompass heterogeneous environments using agentic AI. Agents are self-motivated and self-directed decision-makers that take intelligent action without the need for external triggers [2, 4]. An agentic perceptron module was introduced to drive the coordinated orchestration of intelligent action and self-motivation.

g. 4. Policy-Governance Compliance Coverage by Framework (%

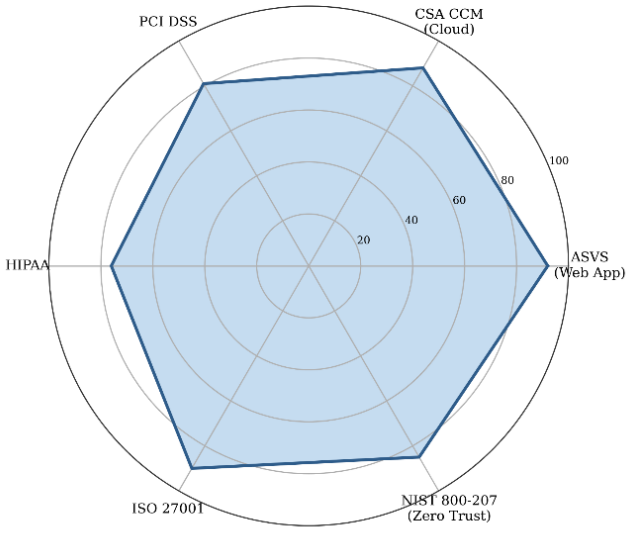

Fig. 4. **Compliance framework coverage (radar)**: coverage across ASVS, CSA CCM, PCI DSS, HIPAA, ISO 27001, and NIST SP 800-207 Zero Trust, computed with the weighted measure of (12) and discussed in Section 4-A.

Agentic AI instantiates decision and action modules to support security orchestration across Oracle Cloud and external multi-clouds. Security governance for cloud and multi-cloud platforms, policy administration modelling ontology, and decision-module integration are treated jointly. Heterogeneous environments require heterogeneous action decision-making with external support across cloud platforms. Agentic AI-based security orchestration

synthesizes decision, action, and governance modules to address cloud and multi-cloud security considerations encompassing cloud-native, non-cloud-native, and internal or external environments.

A. Learning and Adaptation

Decision and action modules are not self-contained. Different implementations operate in various environments and evolve as network emulators and exercisers learn from data scrapers, tenant change-notification services, and external threat-intelligence feeds, sharing knowledge with all agents and horizons. The formal mechanism of that sharing is the recursive belief update of (3) combined with the value update of (10).

Table 1. Core Components of Agentic AI-Based Security Orchestration

Component	Description	Main Function	Expected Outcome
Perception Module	Collects contextual data from cloud, network, application, and user sources	Security data observation and environment awareness	Unified security visibility
Data Ingestion Layer	Receives events from SIEM, SOAR, sensors, and cloud-native tools	Converts security data into usable formats	Structured security intelligence
Decision Module	Uses AI/ML models for anomaly detection, classification, and prediction	Determines threat severity and response priority	Risk-aware decision-making
Action Module	Executes workflows, enforcement rules, and remediation actions	Automates security response	Faster incident mitigation
Governance Module	Evaluates candidate actions against policy-as-code	Compliance validation and audit evidence	Provable regulatory alignment

4. Theoretical Foundations of Agentic Security Orchestration

Formal definitions of agentic security orchestration establish a process of security analysis, decision-making, and specific resource action, tailored for Oracle Cloud and extensible to multi-cloud platforms. The necessity of dynamic actions to mitigate recognised attacks and their correlations leads to a layered agentic system that automates knowledge acquisition and autonomy, synchronised with learning through reinforcement and a catalogue of plans and responses. An online learning approach applies policy governance to security decisions and adaptive orchestrator actions. An agent-oriented paradigm for the architecture of security systems enables a multi-layer structure that integrates a threat-intelligence support module with the monitoring and decision-making functionalities of an adaptive orchestrator, with action planning included.

The formal definition of agentic security orchestration synthesizes a process that encompasses decision, action, and policy governance within a dynamic analysis focused on security incident detection and on the corresponding mitigation measures. Security incidents involve recognised attacks whose detection leads a scorer to select a query with growing values. These queries also signify attack correlations, the detection of which paves the way for subsequent actions that mitigate incident consequences.

Table 2. Mapping of Security Functions to Cloud Security Operations

Security Function	Source / Input	Processing Method	Output / Action
Threat Detection	SIEM alerts, logs, sensors	Anomaly detection and classification, (4)–(5)	Threat identification

Vulnerability Validation	CVE data, red-team simulations	Security testing and attack emulation, Algorithm 4	Confirmed vulnerability status
Risk Scoring	ASVS, CSA CCM, PCI DSS, HIPAA, ISO 27001	Rule-based and AI-supported scoring, (2)	Prioritized risk level
Policy Enforcement	Policy-as-code rules	Governance validation, (8)	Approved or blocked action
Incident Response	SOAR playbooks	Automated workflow execution, Algorithm 1	Remediation or containment
Compliance Monitoring	Audit logs, policy checks	Continuous validation, (12)	Compliance evidence

A. Policy Governance and Compliance

Policies govern risk-based management during normal operation and oversight during incident response [16]. The knowledge module reasons about security policies and ensures compliance during learning and adaptation through a multi-layer policy structure supporting inherent policy monitoring and validation. As functions delegating policy definitions, the detection and reaction modules rely on the formal structure of monitoring policy-based compliance. Mission and operational capabilities are defined in conjunctive normal form at one level of the hierarchy; system capabilities and their protection constitute a second layer; and deployment and operational conditions are modelled in a third layer, with logical implications encoded, monitored, and validated by the knowledge module. The satisfaction of that hierarchy is scored by (12), and the floor enters the orchestration objective as the hard constraint in (14). Γmin

Table 3. Decision and Action Module Responsibilities

Module Type	Key Responsibility	Techniques Used	Security Benefit
Decision Module	Analyze security events and generate situational awareness	Deep learning, anomaly detection, prediction	Early threat recognition
Decision Module	Classify incidents and identify causes	Decision trees, contextual reasoning	Accurate incident understanding
Decision Module	Prioritize vulnerabilities	Risk scoring and compliance mapping	Better remediation planning
Action Module	Trigger remediation workflows	SOAR automation	Reduced response delay
Action Module	Apply enforcement rules	Policy-as-code	Consistent security control
Action Module	Generate SOPs and evidence	Automated documentation	Improved audit readiness

A central element of decision-making and action selection is incident classification and cause identification. An agentic endeavour rests on the nature of allocated capabilities, with implied social properties leveraged to address recognition and reaction as evidenced by direct engagements supporting algorithms that focus on local perceptions and service-level knowledge. Recognised classification and cause are the results of incident detection and system reaction supported by knowledge of environment interactions; yet for effective incident reaction, inference of causes — which can be deeper than what is captured by structure — remains a requisite to address resilience. Decision trees formalise recognition and hence learning, with education concerning cross-environment and multi-organisational engagements stipulated by operant-situational levels governing capabilities and classified as human or friendly actors, agents, and hostile elements.

5. Algorithmic Realization

The orchestration behaviour formalised in Section 2 is realised by four cooperating procedures. Algorithm 1 is the outer control loop; Algorithm 2 is the compliance gate invoked before any state-changing action; Algorithm 3 is the offline adaptation of the response policy; and Algorithm 4 is the adversary-emulation procedure through which control efficacy ε_i in (15) is measured rather than assumed.

Algorithm 1: Agentic Multi-Cloud Detection–Decision–Action Loop

Input: event streams $\mathcal{S}$ from OCI, Azure, AWS, and on-premises sensors; policy set P_t; action catalogue $\mathcal{A}$; risk threshold τ

Output: executed action set $\mathcal{X}_t$; updated policy P_{t+1}; audit record Λ_t

initialize $Q \leftarrow Q_0$, $\Lambda_t \leftarrow \emptyset$, $\mathcal{X}_t \leftarrow \emptyset$

for each orchestration epoch $t = 1,2, \ldots$ **do**

  $E_t \leftarrow \text{Normalize}\big(\text{Ingest}(\mathcal{S})\big)$ *// Eq. (1)*

  $C_t \leftarrow \text{ResolveContext}(E_t)$ *// asset criticality, blast radius*

  $\hat{X}_t \leftarrow \text{PredictNormal}(X_{t-1}, \ldots, X_{t-k})$

  $AS_t \leftarrow \| X_t - \hat{X}_t \|_2$ *// Eq. (4)*

  for each candidate threat i derived from E_t **do**

    $P_i \leftarrow \text{BayesUpdate}(P_i, e_i)$ *// Eq. (3)*

    $R_i \leftarrow P_i \times I_i$ *// Eq. (2)*

  end for

  $\mathcal{T}_t \leftarrow \{\, i : R_i \geq \tau \ \vee \ AS_t \geq \tau_a \,\}$

  if $\mathcal{T}_t = \emptyset$ **then continue**

  $\mathcal{T}_t \leftarrow \text{SortDescending}(\mathcal{T}_t, R)$

  for each threat $i \in \mathcal{T}_t$ **do**

    $Y_i \leftarrow \text{argmax}_k P(y_k \mid X_i)$ *// Eq. (5)*

    $D_t \leftarrow f(E_t, C_t, P_t)$ *// Eq. (6)*

    $A^* \leftarrow \text{argmax}_{A \in \mathcal{A}} U(A \mid D_t)$ *// Eq. (7)*

    $\big(V_p, \lambda\big) \leftarrow \text{ValidatePolicy}(A^*, P_t)$ *// Algorithm 2*

    if $V_p = 1$ **then**

      $o_i \leftarrow \text{Execute}(A^*)$; $\mathcal{X}_t \leftarrow \mathcal{X}_t \cup \{A^*\}$

    else

      $o_i \leftarrow \text{EscalateToAnalyst}(D_t, \lambda)$

    end if

    $r \leftarrow \text{Reward}(o_i, R_i, \Delta\text{MTTR})$

    $Q(s,a) \leftarrow Q(s,a) + \eta[\, r + \gamma \max_{a'} Q(s', a') - Q(s,a) \,]$ *// Eq. (10)*

    $\Lambda_t \leftarrow \Lambda_t \cup \langle e_i, R_i, D_t, A^*, V_p, o_i \rangle$

  end for

  $\Gamma \leftarrow \text{ComplianceCoverage}(\Lambda_t, F)$ *// Eq. (12)*

  if $\Gamma \geq \Gamma_{\min}$ **then** $P_{t+1} \leftarrow P_t + \Delta P(R_t, F_t)$ *// Eq. (11)*

  else $P_{t+1} \leftarrow P_t$; **raise** governance exception

31: **end for**

32: **return** $\mathcal{X}_t$, P_{t+1}, Λ_t

Algorithm 1 executes in $O(|E_t| + |\mathcal{T}_t| \log |\mathcal{T}_t| + |\mathcal{T}_t| \cdot |\mathcal{A}|)$ per epoch, the dominant terms being the sort at line 13 and the utility maximisation at line 17. Two design decisions deserve emphasis. Line 12 makes the loop vacuous when no threat crosses the risk floor, which is the ordinary case and keeps steady-state cost proportional to ingestion rather than to catalogue size. Line 30 is the governance interlock: when coverage falls below $\Gamma_{\min}$ the orchestrator freezes its own policy rather than continuing to adapt, so that a degraded compliance posture cannot be normalised by the learning process itself.

Algorithm 2: Policy-as-Code Validation with Compliance-Gated Execution

Input: candidate action A^{*}; layered policy set $P_t = \langle P^{(1)}, P^{(2)}, P^{(3)} \rangle$

Output: validation flag V_p; violated-clause set λ

1: $\lambda \leftarrow \emptyset$

2: **for** each layer $\ell = 1$ **to** 3 **do** *// mission, capability, deployment*

  3: **for** each clause $c \in P^{(\ell)}$ in conjunctive normal form **do**

    4: **if** $\neg\, \mathrm{Entails}(A^{*}, c)$ **then** $\lambda \leftarrow \lambda \cup \{c\}$

  5: **end for**

  6: **if** $\lambda \neq \emptyset$ **and** $\ell = 1$ **then break** *// mission clauses are non-negotiable*

7: **end for**

8: **if** $\lambda = \emptyset$ **then**

  9: $V_p \leftarrow 1$; **emit** signed evidence record for A^{*} *// Eq. (8)*

10: **else**

  11: $V_p \leftarrow 0$

  12: **if** $\mathrm{HasCompensatingControl}(\lambda)$ **then**

    13: $A^{*} \leftarrow \mathrm{Substitute}(A^{*}, \lambda)$; **goto** 1 *// bounded to κ retries*

  14: **end if**

15: **end if**

16: **return** V_p, λ

Validation proceeds outward from the most constraining layer, so an action that violates a mission-level clause is rejected without evaluating the cheaper deployment-level clauses. The substitution branch at line 13 is bounded to κ retries, which guarantees termination and prevents the orchestrator from searching indefinitely for a policy-satisfying variant of an action that is fundamentally disallowed.

Algorithm 3: Reinforcement-Learning Playbook Adaptation

Input: experience buffer $\mathcal{B} = \{(s, a, r, s')\}$; learning rate η; discount γ; exploration schedule ϵ_t; validation set $\mathcal{V}$

Output: adapted playbook policy π^{*}

1: **initialize** Q from the incumbent playbook; $J^{*} \leftarrow \infty$

2: **repeat**

  3: sample minibatch $\mathcal{M} \sim \mathcal{B}$ with priority $\propto |\delta|$

  4: **for** each $(s, a, r, s') \in \mathcal{M}$ **do**

    5: $\delta \leftarrow r + \gamma \max_{a'} Q(s', a') - Q(s, a)$

    6: $Q(s, a) \leftarrow Q(s, a) + \eta\, \delta$ *// Eq. (10)*

7: **end for**

8: $\pi \leftarrow \epsilon_t$-greedy$(Q)$

9: evaluate $J(\pi) = w_1\text{MTTD} + w_2\text{MTTR} + w_3\text{FPR}$ on $\mathcal{V}$ *// Eq. (14)*

10: **if** $\Gamma(\pi) \geq \Gamma_{\min}$ **and** $J(\pi) < J^*$ **then**

11: $J^* \leftarrow J(\pi)$; $\pi^* \leftarrow \pi$

12: **end if**

13: **until** J^* improves by less than ϵ over w successive evaluations

14: **return** π^*

Adaptation is deliberately offline and candidate-gated. A policy is promoted only when it both improves the composite objective and preserves compliance coverage, so the acceptance test at line 10 encodes the constrained optimisation of (14) directly. Prioritised sampling at line 3 concentrates updates on transitions with large temporal-difference error, which in security telemetry corresponds to rare containment failures — precisely the events from which the playbook has most to learn and which uniform sampling would drown out.

Algorithm 4: Adversary-Emulation Assessment for Control Validation

Input: CVE corpus $\mathcal{C}$; deployed control set $\mathcal{K}$; target environment $\mathcal{E}$; emulation budget B

Output: measured efficacy vector ε; control gap report $\mathcal{G}$

1: $\mathcal{G} \leftarrow \emptyset$; $\varepsilon \leftarrow \mathbf{0}$

2: $\mathcal{C}' \leftarrow \text{SelectApplicable}(\mathcal{C}, \mathcal{E})$ *// reachability and version filter*

3: $\mathcal{C}' \leftarrow \text{RankByExploitability}(\mathcal{C}')$; truncate to budget B

4: **for** each technique $\mu \in \mathcal{C}'$ **do**

5: RedTeam: instantiate μ against an isolated replica of $\mathcal{E}$

6: $d \leftarrow \mathbb{1}[\,\text{Detected}(\mu)\,]$; $b \leftarrow \mathbb{1}[\,\text{Blocked}(\mu)\,]$

7: $\varepsilon_\mu \leftarrow \text{EfficacyScore}\left(d, b, t_\mu^{\text{det}}\right)$

8: **if** $d = 0$ **or** $b = 0$ **then**

9: $\mathcal{G} \leftarrow \mathcal{G} \cup \langle \mu, \text{MapToControl}(\mu, \mathcal{K}) \rangle$

10: BlueTeam: synthesize detection rule and candidate policy clause for μ

11: **end if**

12: **end for**

13: **submit** $\mathcal{G}$ to the learning and adaptation module as F_t *// feeds Eq. (11)*

14: **return** $\varepsilon, \mathcal{G}$

Algorithm 4 closes the loop between assumed and demonstrated protection. Without it, the efficacy term ε_i in (15) is an estimate supplied by the control vendor; with it, that term is measured against the deployed configuration, and gaps are returned to the adaptation module as threat-intelligence feedback F_t. The reachability filter at line 2 matters operationally: unfiltered CVE corpora are dominated by entries that cannot be instantiated in the target environment, and emulating them consumes budget without yielding efficacy information.

6. Architecture and Components

The architecture encompasses distinct modules for perception, decision, action, and policy management, seamlessly integrated within a single agentic orchestrator. Multi-layered, AI-backed perception capabilities gather security-related contextual data from internal systems and contextualised information from external sources [24, 35]. Data is ingested for processing by heterogeneous specialist tools, with outputs translated for orchestrator consumption before being forwarded to the appropriate decision modules. Foundations for agentic AI-based security orchestration are provided, as are essential building blocks for the practical implementation of the architecture.

Table 4. Agentic AI Security Orchestration Workflow

Step	Process	Description	Result
1	Data Collection	Security data is collected from cloud-native tools, networks, applications, and external feeds	Raw security context
2	Event Ingestion	SIEM/SOAR platforms ingest and normalize security events, (1)	Machine-readable inputs
3	Threat Analysis	Decision agents analyze anomalies, vulnerabilities, and attack patterns, (3)–(5)	Security insight
4	Risk Evaluation	Incidents are scored using security standards and organizational policies, (2)	Risk priority
5	Policy Validation	Proposed actions are checked against governance and compliance rules, Algorithm 2	Approved response path
6	Automated Response	SOAR playbooks and action agents execute remediation, Algorithm 1	Threat containment
7	Learning Update	Logs and outcomes are fed back into learning modules, (10)–(11)	Improved future decisions

Following perception, security decision modules determine the need for security interventions. When required, an agentic decision module prepares high-level, evidence-supported security interventions that respect determined corporate policies, compliance rules, and constraints. Such interventions — or the need to reduce actor entities to lower risk — provide the triggers for action modules. Rich communication features among agents allow for negotiation and, where possible, consensus-based solutions and recommendations. Finally, the outcome is always checked against compliance and policy requirements for a record of implementation governance.

A. Perception and Data Ingestion

Applications, network devices, facilities, services, and users produce large volumes of data that drive perceptions of the security state by the security orchestration system. The volume of data may also provide the basis for self-calibration, adaptation, replacement, or mitigation actions. Security events are detected by SIEM or SOAR platforms and ingested by decision modules, while a large language model governs learning and adaptation by service component decision modules.

The system requires central perception of all data sources in the Oracle Public Cloud and any on-premises components in the Oracle Private Cloud at Customer environment, so that decision nodes in each layer have synchronised contexts. Perception and data ingestion connect percept data sources with dedicated add-in mechanisms for data extraction and present their outputs in consumer-optimised formats. Detecting security events and ingesting structured data in machine-readable formats are not considered perception components, because such activities are already performed by SIEM and SOAR platforms.

Table 5. Multi-Cloud Security Governance Requirements

Governance Area	Requirement	Implementation Method	Supported Environment
Identity and Access Control	Enforce secure access permissions	IAM policy automation	OCI and multi-cloud
Compliance Assurance	Maintain regulatory alignment	Policy-as-code and audit checks, (12)	PCI DSS, HIPAA, ISO 27001
Threat Response	Automate containment and remediation	SOAR workflows, Algorithm 1	Public, private, and hybrid cloud

Vulnerability Management	Detect and reduce exposure	CVE validation and red-team testing, Algorithm 4	PaaS and cloud-native systems
Audit Readiness	Preserve response evidence	Automated SOP and evidence generation	Enterprise security operations
Zero Trust Alignment	Validate every access and action	Continuous policy validation, (8)	Distributed cloud infrastructure

7. Empirical Evidence from Recent Deployment Data

The preceding sections argue for agentic orchestration on structural grounds. This section tests that argument against independently collected industry telemetry from 2025, which is informative precisely because it aggregates outcomes across organisations at very different levels of automation maturity rather than reporting a single vendor deployment.

A. The Breach Lifecycle Is Contracting

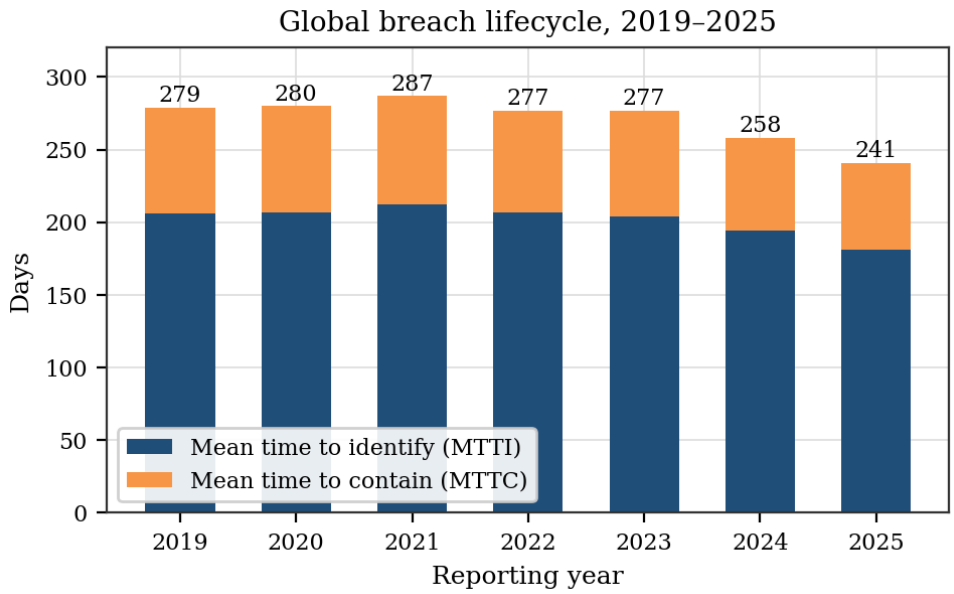

Fig. 5. Global mean time to identify (MTTI) and mean time to contain (MTTC) a breach, 2019–2025. Totals correspond to as defined in (13). MTTD+MTTR

Fig. 5 plots the two components of the breach lifecycle over seven reporting years. The combined figure fell from a peak of 287 days in 2021 to 241 days in 2025 — a nine-year low — with identification falling from 212 to 181 days and containment from 75 to 60 days. Two features of the trend matter for the present argument. First, the decline is monotone from 2021 onward rather than a single-year artefact, which is consistent with a structural change in tooling rather than with year-to-year variation in the incident mix. Second, identification and containment are improving at different rates: identification has fallen by roughly 15% since 2021 while containment has fallen by 20%, indicating that automation has been absorbed more readily into the response path than into the detection path. This is the asymmetry the orchestration objective in (14) is designed to expose, since a weight vector that penalises only will drive investment toward the component that is already improving faster. MTTR

B. Cost and Dwell Time Scale with Environmental Heterogeneity

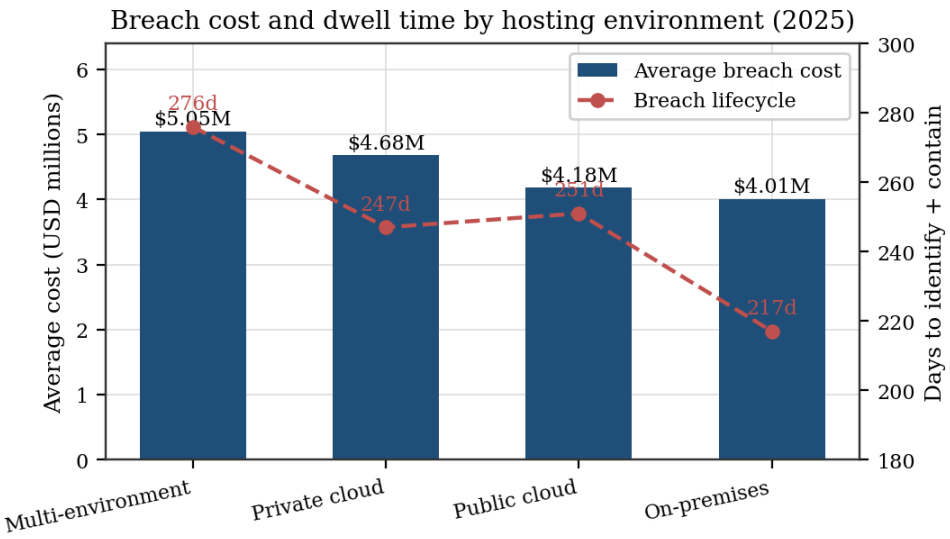

Fig. 6. Average breach cost and breach lifecycle by hosting environment, 2025. Multi-environment estates carry both the highest cost and the longest dwell time.

Fig. 6 is the most direct empirical support for a multi-cloud orchestration layer. Breaches involving data spread across multiple environments cost an average of USD 5.05 million and took 276 days to identify and contain, against USD 4.01 million and 217 days for wholly on-premises incidents: a 26% cost premium and 59 additional days of exposure. Private and public cloud estates fall between these extremes, at USD 4.68 million over 247 days and USD 4.18 million over 251 days respectively. The ordering is instructive because it does not track cloud adoption as such — public cloud is the *cheapest* of the three cloud configurations — but rather tracks fragmentation of the observation surface. An estate distributed across providers presents the defender with several partial views and no common lens, which is precisely the condition that the unified event vector of (1) and the context-resolution step at line 4 of Algorithm 1 exist to remedy. The 59-day gap can therefore be read as the empirical cost of the correlation work that an orchestration layer performs automatically and that, in its absence, is performed manually or not at all.

C. Automation Depth, Not Automation Presence, Drives the Benefit

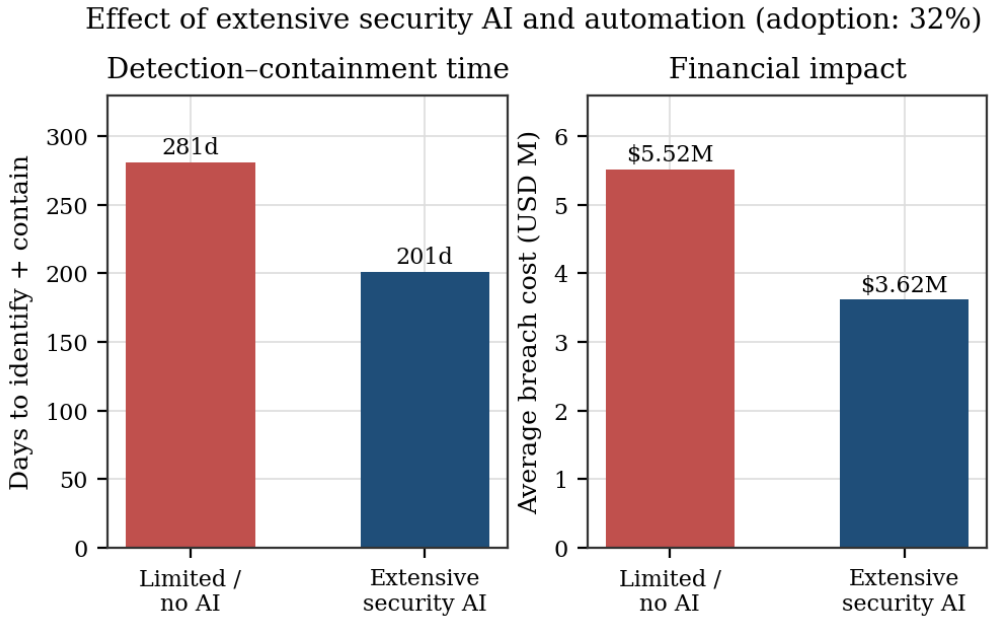

Fig. 7. Detection–containment time and financial impact for organisations making extensive use of security AI and automation, against those making limited or no use. Extensive adoption stood at 32% of surveyed organisations in 2025.

Fig. 7 separates organisations by the depth of their security automation. Those deploying AI and automation extensively identified and contained breaches roughly 80 days faster and incurred approximately USD 1.9 million less in breach cost than those with limited or no deployment. The distribution behind this average is as important as the average itself: only 32% of surveyed organisations fell into the extensive-use category, so the aggregate improvement visible in Fig. 5 is being produced by a minority of the population. Three observations follow. First, the marginal value of orchestration is largest for the 68% who have not yet reached extensive use, which argues against treating the aggregate trend as evidence of saturation. Second, the benefit is realised in both terms of (14) simultaneously — time and cost move together — which supports the use of a single composite objective rather than separately optimised detection and response programmes. Third, the gap is consistent with the residual-risk decomposition in (15): organisations with shallow automation have low because correct actions are selected inconsistently under analyst load, and the product therefore remains small even where individual controls are well configured. ριριεi

D. Governance Debt as the Limiting Factor

The same 2025 evidence records a governance shortfall that bears directly on the constraint in (14): 63% of organisations reported having no formal AI governance policy, and only 34% conducted regular audits capable of detecting unsanctioned AI use. Breaches involving such unsanctioned deployments averaged USD 4.63 million, approximately USD 670,000 above the baseline incident. Supply-chain compromise accounted for roughly 15% of breaches at USD 4.91 million and 267 days, the longest dwell time of any vector measured. These figures are the reason the compliance gate in Algorithm 2 is placed before execution rather than after, and the reason the governance interlock at line 30 of Algorithm 1 freezes policy adaptation when coverage falls below . An orchestrator that accelerates response while its governance coverage decays converts a detection problem into an accountability problem, and the cost data indicate that the second is more expensive than the first. Γmin

Table 6. Summary of 2025 Empirical Indicators and Their Formal Counterparts

Indicator	Reported value (2025)	Formal counterpart
Global breach lifecycle	241 days (181 identify + 60 contain)	, (13)
Multi-environment breach cost	USD 5.05 M over 276 days	Fragmented , (1)
Extensive-automation advantage	~80 days and ~USD 1.9 M	term, (15)
Extensive-automation adoption	32% of organisations	Enforcement score , (9)
Organisations without AI governance policy	63%	Coverage floor , (12), (14)
Supply-chain compromise dwell time	267 days	Cross-tenant belief update, (3)

8. Discussion and Threats to Validity

Three limitations qualify the conclusions above. The empirical indicators in Section 7 are survey-derived aggregates across industries and geographies; they establish association between automation depth and outcome, not causation, and organisations able to fund extensive automation may differ systematically in ways that independently improve their security posture. The formulation in Section 2 assumes that impact severity in (2) is known and stable, whereas in practice it is elicited from asset owners and drifts as workloads are re-platformed; a mis-specified propagates directly into the triage order at line 13 of Algorithm 1. Finally, the reward signal at line 24 is derived from observed containment, which is itself a function of detection: threats that are never detected contribute no negative reward, so the value function of (10) is estimated on a censored sample. Adversary emulation (Algorithm 4) partially corrects this censoring by generating detections for techniques the production pipeline missed, which is a further argument for treating emulation as a continuous control rather than a periodic audit. IiIi

9. Conclusion

Data protection against different incident types is achieved by a single system, covering components initially dispersed and semi-automated. Evidence-based actions align cybersecurity protocols with operational policies. Cloud-native systems and AI-based orchestration improve efficiency without burdening information security [6, 15]. Learning modules adapt countermeasure settings to incident-landscape dynamism, reducing the number of incidents and damage costs. Policy control ensures compliance with established security governance and controls expansion into a multi-cloud environment. The application of AI throughout cloud security orchestration enables a more complete approach, simultaneously taking actions and suggesting security-oriented decisions.

This extended treatment adds three elements to that proposal. The orchestration loop is now specified as a system of fifteen equations in which risk estimation, decision, compliance validation, and adaptation are mutually referencing rather than independently defined, so that the effect of a change in any one term can be traced through the rest. Four algorithms render that specification executable, with explicit termination conditions, complexity characteristics, and a governance interlock that prevents the learning process from relaxing the constraints it operates under. The empirical section anchors the argument in 2025 telemetry showing a 241-day global breach lifecycle, a 26% cost premium for multi-environment estates, and an approximately 80-day and USD 1.9-million advantage for organisations with extensive security automation — a group that still comprises fewer than a third of organisations. Taken together, these results suggest that the binding constraint on agentic security orchestration is no longer detection capability but governance coverage, and that future work should concentrate on making compliance evidence a machine-verifiable output of the orchestration loop rather than a periodic reconstruction of it.

References

1. Loganathan, R., Amistapuram, K., & Aitha, A. R. (2026). Letter to the Editor re Annual Updates of the European Association of Urology–European Society for Pediatric Urology (EAU-ESPU) Paediatric Urology Guidelines: Are Large-Language Models (LLM) Better Than the Usual Structured Methodology? Journal of Pediatric Urology.
2. Bandi, A., Kongari, B., Naguru, R., Pasnoor, S., & Vilipala, S. V. (2025). The rise of agentic AI: A review of definitions, frameworks, architectures, applications, evaluation metrics, and challenges. Future Internet, 17(9).
3. Recharla, M., Garapati, R. S., Bandi, V. D. V. K., Yandamuri, U. S., & Mangalampalli, B. M. (2026, March). Generative AI-Enhanced Data Engineering Pipelines for Predictive Biomarker Discovery in Alzheimer's and Kidney Disease. In 2026 IEEE International Conference on AI Engineering and Innovations (AIEI) (pp. 1–5). IEEE.
4. Huang, K. (2025). Agentic AI governance and secure autonomous systems. O'Reilly Media.
5. Kummari, D. N., Aitha, A. R., Kumar, M. V. K., Pandiri, L., Gottimukkala, V. R. R., & Nagubandi, A. R. (2026, March). Real-Time AI-Driven Audit and Compliance Framework for Smart Manufacturing Financial Workflow. In 2026 IEEE International Conference on AI Engineering and Innovations (AIEI) (pp. 1–5). IEEE.
6. Shua, A., & Carpenter, N. (2024). AI-driven cloud security automation for cloud-native application protection. Journal of Cloud Computing, 13(1), 78–96.
7. Scarfone, K., & Souppaya, M. (2024). Guidelines on security and privacy in public cloud computing. National Institute of Standards and Technology.
8. Amistapuram, K., Kolla, T., Bandi, V. D. V. K., Kolla, S. K., & Rani, P. S. Journal of Rare Cardiovascular Diseases.
9. Humble, J., & Farley, D. (2023). Continuous delivery: Reliable software releases through build, test, and deployment automation (2nd ed.). Addison-Wesley.
10. Designing Autonomous LLM Agent Frameworks Using Gen AI Pipelines to Enhance Customer Service Management and Knowledge Workflows. (2025). Lex Localis — Journal of Local Self-Government, 23(S6), 9719–9733.
11. Burns, B., Beda, J., Hightower, K., & Evenson, L. (2023). Kubernetes: Up and running (3rd ed.). O'Reilly Media.
12. Mangala, N., Amistapuram, K., & Garapati, R. S. (2026). Comment on "Explainable machine learning on clinical features to predict and differentiate Alzheimer's progression by sex: Toward a clinician-tailored web interface". Journal of the Neurological Sciences, 488.
13. Richards, J., & Smith, M. (2023). Autonomous cyber defense using reinforcement learning for cloud environments. IEEE Access, 11, 114210–114228.
14. Bhavani, B. D., SR, S., Loganathan, R., & Nagaraj, S. (2026, April). Evolutionary Gravitational Neocognitron Neural Network, Snow Leopard Optimization and Deep Graph Reinforcement Learning for Routing Protocol in WSN. In 2026 2nd International Conference on Intelligent Systems and Computational Networks (ICISCN) (pp. 1–8). IEEE.
15. Sharma, P., Chen, L., & Wang, X. (2023). Multi-cloud security management using AI-based orchestration. Future Generation Computer Systems, 143, 156–170.
16. Rose, S., Borchert, O., Mitchell, S., & Connelly, S. (2022). Zero Trust Architecture (NIST Special Publication 800-207). National Institute of Standards and Technology.
17. Peddi, R. K. (2024). AI-Based Workforce Analytics for SLA Governance and Uptime Assurance in Data Centers. Journal of Computational Analysis and Applications (JoCAAA), 33(08), 8589–8601.
18. Stallings, W. (2022). Network security essentials: Applications and standards (7th ed.). Pearson.

19. Mattaparthi, R. (2023). Connected Fleet Intelligence: Edge-Centric Analytics and Computer Vision for Predictive Manufacturing and Asset Resilience. International Journal of Advanced Research in Computer Science & Technology (IJARCST), 6(5), 9077–9088.
20. Richards, J., Ford, M., & Sullivan, R. (2022). Artificial intelligence for security orchestration and automated response. Computers & Security, 118, 102725.
21. Radha, S., Gottimukkala, V. R. R., Thottara, S., Vandhana, K., & Gokulraj, J. (2025). Adaptive Video Streaming Over 5G Networks Using Deep Reinforcement Learning with Closed-Loop Feedback Mechanism for Bitrate Control. In 2025 International Conference on Communication, Computer, and Information Technology (IC3IT) (pp. 1–6). IEEE.
22. Chandramouli, R., Butcher, Z., & Ekrann, H. (2022). Security strategies for microservices-based applications. National Institute of Standards and Technology.
23. Kumar, S. S., Garapati, R. S., Segireddy, A. R., Kalisetty, S., Inala, R., & Nagabhyru, K. C. (2026, March). Hybrid Deep Neural Network–DevOps Pipeline Optimization for Risk Prediction in Cloud-Native Workers' Compensation Platforms. In 2026 IEEE International Conference for Convergence in Computing Technology (I3CTCON) (pp. 1–6). IEEE.
24. Casola, V., De Benedictis, A., Rak, M., & Villano, U. (2021). Security-by-design approaches for cloud-native applications. Future Generation Computer Systems, 122, 148–161.
25. Kolla, S. K., Bandi, V. D. V. K., & Meda, R. (2026). Comment on "Predicting self-image satisfaction after adult spinal deformity surgery: a machine learning approach using patient phenotypes". Spine Deformity, 1–3.
26. Hassan, W., Guizani, M., & Al-Fuqaha, A. (2021). Security challenges in cloud-native computing: A survey. IEEE Communications Surveys & Tutorials, 23(4), 2354–2391.
27. Mangalampalli, B. M. (2024). Transparent Intelligence Explainability Frameworks for AI-Driven Clinical Decision Support in Healthcare Business Intelligence. International Journal of Research Publications in Engineering, Technology and Management (IJRPETM), 7(3), 10566–10579.
28. Allaire, J., & Fowler, M. (2021). Cloud native transformation. Addison-Wesley.
29. Isukapatla, M., Davuluri, P. N., Satya, G. S., & Prakash, P. R. (2026, April). An Attention-Enhanced YOLO Framework with IMU Confidence for Road Surface Defect Analysis. In 2026 2nd International Conference on Intelligent Systems and Computational Networks (ICISCN) (pp. 1–8). IEEE.
30. Humble, J., & Molesky, J. (2021). Lean enterprise (2nd ed.). O'Reilly Media.
31. Skelton, M., & Pais, M. (2020). Team Topologies: Organizing business and technology teams for fast flow. IT Revolution.
32. Paramasivam, R., Reddy, S. S., Reddy, V. A. R., Sandra, K., & Narayana, M. V. S. (2026). JellyFish Search Optimization with Arctic Puffin Optimization Algorithm for Efficient Routing Based on the Software Defined Communication Network. In 2026 2nd International Conference on Intelligent Systems and Computational Networks (ICISCN) (pp. 1–6). IEEE.
33. Burns, B., Grant, B., Oppenheimer, D., Brewer, E., & Wilkes, J. (2020). Borg, Omega, and Kubernetes. Communications of the ACM, 63(3), 70–79.
34. Rao, C. S., Bandi, V. D. V. K., Brahmandam, L. M. K., Gantikota, S., & Kannan, K. (2026, April). Topology-Aware Hypergraph Neural Network Framework for Intelligent Decision Support Systems in Network Operations. In 2026 2nd International Conference on Intelligent Systems and Computational Networks (ICISCN) (pp. 1–7). IEEE.
35. Fernandez, E. B., & Monge, R. (2020). Security patterns for cloud systems. IEEE Software, 37(2), 58–66.
36. Krishnan, M., Bandi, V. D. V. K., Mangala, N., Kolla, S. H., & Mangalampalli, B. M. Engineering Intelligent Cloud-Native Data Ecosystems for Predictive Decision-Making in Industry.
37. IBM Security & Ponemon Institute. (2025). Cost of a Data Breach Report 2025: The AI oversight gap. IBM Corporation.
38. Ponemon Institute. (2025). Breach lifecycle and security automation benchmarks, 2019–2025. In Cost of a Data Breach Report 2025. IBM Corporation.