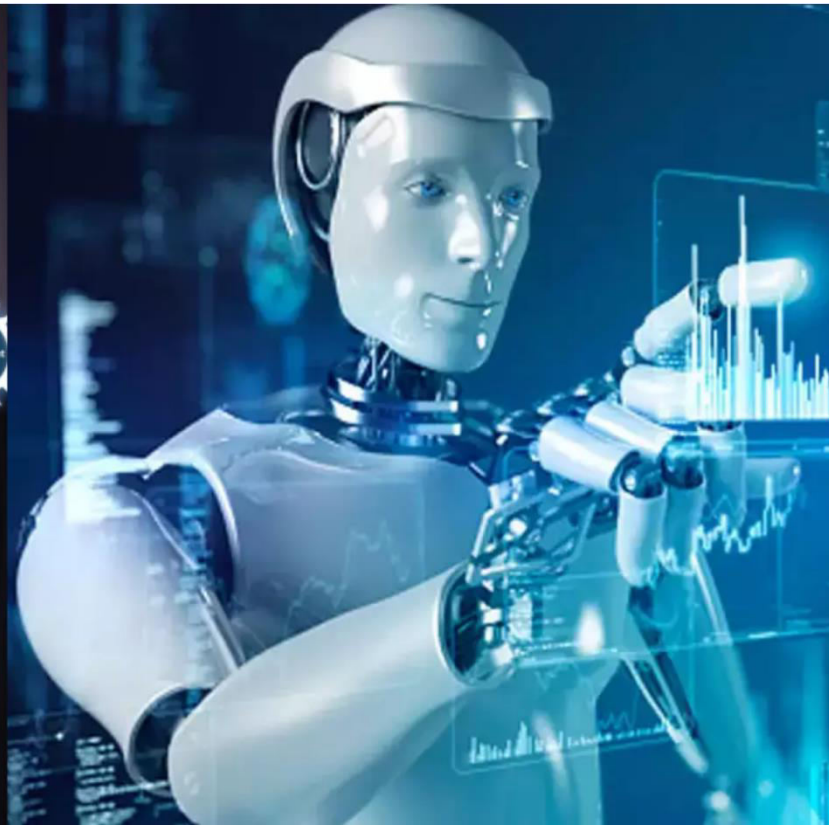
IJIRSET
International Journal of Innovative Research in Science | Engineering | Technology

ISSN(O): 2319-8753
ISSN(P): 2347-6710

International Journal of Innovative Research in Science Engineering and Technology (IJIRSET)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)



Impact Factor: 8.699

Volume 15, Issue 2, February 2026

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 8.699

|www.ijirset.com |A Monthly, Peer Reviewed & Refereed Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|

Volume 15, Issue 2, February 2026

|DOI: 10.15680/IJIRSET.2026.1502103|

Cloud-Native Honeypot for Automated Threat Intelligence

Akruti Pandwal, Chirag Singh Rothan, Priyanshu Vyas

Asst. Professor, Department of Computer Science & Engineering, Parul Institute of Engineering and Technology, Parul University, Vadodara, Gujarat, India

Department of Computer Science and Engineering, Parul University, Vadodara, India

Department of Computer Science and Engineering, Parul University, Vadodara, India

ABSTRACT: The rise in advanced cyber attacks against the cloud infrastructure makes it necessary to have proactive and adaptive security measures. This paper presents the design, implementation and evaluation of a cloud native honeypot system designed for automated threat intelligence generation. The proposed architecture uses containerisation technologies, microservices orchestration and machine learning-based log analysis for the deployment, management and analysis of decoy services in cloud environments. The system automatically captures adversarial tactics, techniques and procedures, correlates attack patterns and generates actionable threat intelligence feeds in standardised formats. Experimental evaluation shows that the system performed well in delivering the detection and classification of more than 94% of simulated attack vectors in a variety of cloud deployment scenarios with an average alert latency of less than 3.2 seconds. A simulated security incident case study is used to validate the end-to-end pipeline from intrusion detection intelligence dissemination. The results show that cloud-native honeypots offer meaningful increases in organisational threat awareness with low resource overhead, and thus can provide a scalable and cost-effective complement to conventional defensive security controls.

I. INTRODUCTION

The modern-day threat landscape is characterised by increasingly sophisticated, automated and persistent cyber attacks exploiting cloud infrastructure at scale [1]. Organisations shifting workloads to public and hybrid cloud environment are facing an increased attack surface, including poorly configured services, application programming interfaces (APIs) and ephemeral compute instances that pose a challenge to traditional perimeter-based defences [2]. Conventional intrusion detection systems, also known as IDS, and security information and event management (SIEM) platforms - whilst fundamental - largely employ a reactive stance, by detecting the threat only after the signatures or behavioural anomalies have been catalogued [3].

Honeypots - intentionally vulnerable systems set up to attract, detect and investigate malicious activity - have long been recognised as a useful proactive security mechanism [4]. By offering lures to adversaries with their tempting targets, honeypot traps the capture of real data from the attacks that enable the enrichment of threat intelligence, helping define the defensive plan and attribution efforts. However, there are several limitations to traditional honeypot deployments: they are quite heavy to configure manually, are not scalable, produce very large volumes of unstructured logs, and are isolated from automated intelligence pipelines [5].

The coming together of cloud-native technologies, such as containerisation, orchestration platforms, such as Kubernetes, serverless computing and infrastructure-as-code, offers an enticing opportunity to reimagine honeypot architectures [6]. Cloud-based honeypots are quickly deployed and dynamically scaled depending on what they detect in the environment and then decommissioned with no residual risk. Furthermore, combining Machine Learning algorithms (for log classification) and Natural Language Processing (for the extraction of indicators) allows us to transform raw honeypot data into structured actionable threat intelligence automatically [7].

1.1 Research objectives

This research has the following objectives: (i) design a modular and cloud-native honeypot architecture that emulates diverse services across multiple cloud providers (ii) develop an automated pipeline that takes honeypot telemetry as

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 8.699

|www.ijirset.com |A Monthly, Peer Reviewed & Refereed Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|

Volume 15, Issue 2, February 2026

|DOI: 10.15680/IJIRSET.2026.1502103|

input, classifies attack behaviours and generates threat intelligence in Structured Threat Intelligence Expression (STIX) format (iii) evaluate the detection efficacy, classification accuracy and performance overhead of the proposed system with a controlled experimentation and a simulated security incident case study.

1.2 Paper organisation

The rest of this paper is organised as follows. Section 2 is a review of related literature and foundation concepts. Section 3 is the articulation of the problem statement and identification of research gaps. Section 4 describes the threat model and risk assessment. The system architecture and design are described in section 5. The methodology is detailed in section 6. Section 7 is about implementation and testing of Results and discussion are presented in section 8. Experimental evaluation and performance testing are given in section 9. Section 10 presents a simulation security incident case study. Section 11 is about limitations and consideration. Section 12 concludes the paper and Section 13 provides future improvements.

II. LITERATURE REVIEW AND BACKGROUND

2.1 Honeypot taxonomy and evolution

Honeypots are broadly categorized according to the level of interaction. Low-interaction honeypots, such as Honeyd [4], mimic services at the network protocol level, allowing metadata of the connection but not allowing full access to the system. Whilst resource-efficient they provide limited insight to post-exploitation behaviour. High-interaction honeypots, such as those deployed by the Honeynet Project [8], provide full operating system environments, allowing to see lateral movement, privilege escalation and data exfiltration. However, they come with a built-in risk of compromise and require a lot of maintenance.

Medium-interaction honeypots, which include Cowrie [9] and Dionaea [10], represent a middle ground of simulating functionality at the service level (e.g. secure shell (SSH) command execution or server message block (SMB) file transfers) without exposing real system functions. These types of tools have reached a certain popularity due to their balance of fidelity and safety.

2.2 Cloud-native security paradigms

Cloud-native computing, as defined by the Cloud Native Computing Foundation (CNCF) emphasises on containerised, dynamically orchestrated and microservices oriented application architectures [11]. Security in the cloud-native environment presents new challenges such as escaping containers, misconfigurations of orchestration tool and supply chain attacks on container images [12]. Some recent work by Sultan et al. [13] studied container security taxonomies, whilst Shamim et al. [14] studied the attack vectors of Kubernetes.

The use of honeypot concepts in cloud-native environments is still new. Kyriakou and Sklavos [15] have proposed a Docker-based honeypot framework, however, they did not work on an automated intelligence generation. Vasilomanolakis et al. [16] have conducted a survey on honeypot technologies in a comprehensive manner but identified the absence of cloud-native integration as a major gap.

2.3 Threat intelligence automation

Threat intelligence, which can be defined as evidence-based knowledge about existing or emerging threats. [17] Threat intelligence is increasingly being consumed and produced in formats that machines can read. The STIX standard, which is maintained by the Organisation for the Advancement of Structured Information Standards (OASIS), describes a structured language for describing cyber threat information [18]. The Trusted Automated Exchange of Intelligence Information (TAXI) protocol is used to facilitate the automated sharing of such intelligence [19].

Recent developments in machine learning have made it possible to automatically classify network traffic and log data. Li et al. [20] used recurrent neural networks (RNNs) for intrusion detection and had an accuracy of 96% on benchmark datasets. Buczak and Guven [21] conducted a survey of data mining and machine learning techniques for cyber analytics, concluding that ensemble techniques provide powerful generalisation for various attack types.

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 8.699

|www.ijirset.com |A Monthly, Peer Reviewed & Refereed Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|

Volume 15, Issue 2, February 2026

|DOI: 10.15680/IJIRSET.2026.1502103|

2.4 Related systems

Several systems have tried to address the gap between honeypots and threat intelligence. T-Pot developed by Deutsche Telekom [22], bundles several honeypot tools behind a unified management interface and exports data to the Elasticsearch, Logstash and Kibana (ELK) stack. MHN (Modern Honey Network) [23] offers centralised deployment and normalisation of data collection However, neither any of the systems offer native cloud orchestration, automated STIX generation or machine learning for classification, leaving substantial scope for improvement.

III. PROBLEM STATEMENT AND RESEARCH GAP

Despite the individual honeypot tools and the overall sophistication of threat intelligence platforms having come of age, there is a major gap where they meet in the cloud-native world. Specific examples of such deficiencies are given below:

3.1 Lack of cloud-native orchestration

Existing honeypot frameworks are mostly written so that they can be deployed on bare metal or virtualised infrastructure. They do not make use of container orchestration capabilities such as automatic scaling, self-healing, rolling updates and declarative configuration management offered by platforms such as Kubernetes [6]. This reduces their agility and adds to operational overhead.

3.2 *Manual intelligence extraction*

The process of transforming raw honeypot log data into structured threat intelligence data is a mostly manual process that is analysts-heavy. Current tools create voluminous logs which require expert interpretation to identify indicators of compromise (IoCs), attack patterns and adversary tactics correlated to frameworks such as MITRE ATT&CK [24]. This is a bottleneck for the timely dissemination of intelligence.

3.3 Being without integrated classification

Whilst machine learning has been used in a wide range of ways to aid network intrusion detection, its incorporation directly into honeypot data pipelines for classification (real-time attack detection) and behavioural clustering hasn't been researched to the same extent. Existing deployments usually export data to external analytics platforms, which creates latency and architectural complexity.

3.4 Limited scalability evaluation

Performance evaluations of honeypot systems in cloud-native deployment conditions, such as horizontal pod autoscaling, multi-region distribution and variable attack intensity, are rare in the literature. Consequently, practitioners do not have empirical guidance for capacity planning and resource allocation.

This research aims to fill these research gaps by proposing, implementing and rigorously evaluating an integrated cloud native honeypot system with rock-star threat intelligence functionalities.

IV. THREAT MODEL AND RISK ASSESSMENT

4.1 Adversary model

The threat model takes as its adversaries everything from opportunistic automated scanners to focused advanced persistent threat (APT) groups. Three levels of adversaries are defined:

4.1.1 Tier 1 - Automated scanners: These actors use mass scanning tools (e.g. Masscan, Shodan crawlers) to scan for exposed services and exploit known vulnerabilities (e.g. via pre-built exploits - exploits kits). They are the greatest volume and lowest sophistication threat.

4.1.2 Tier 2 - Semi-targeted attackers: These attackers perform reconnaissance to locate specific organisational assets, employ credential stuffing and misconfigurations. They may use tunnelling and encryption to elude detection.

4.1.3 Tier 3 - Advanced persistent threats: Sophisticated actors w/ bespoke tooling, zero-day exploits and operational security awareness. They may be aware of the signs of honeypot and may try to evade it or counter-deceive..

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 8.699

|www.ijirset.com |A Monthly, Peer Reviewed & Refereed Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|

Volume 15, Issue 2, February 2026

|DOI: 10.15680/IJIRSET.2026.1502103|

4.2 Risk assessment

Table summarises the risk assessment for the proposed honeypot deployment.

Table Risk assessment matrix for cloud-native honeypot deployment

4.3 Ethical considerations

Threat	Likelihood	Impact	Mitigation
Honeypot compromise used as pivot	Medium	High	Network isolation, egress filtering
Honeypot fingerprinting by adversary	Medium	Medium	Realistic service emulation, dynamic reconfiguration
Denial of service against honeypot	High	Low	Autoscaling, rate limiting
Data exfiltration of captured intelligence	Low	High	Encryption at rest and in transit
False positive intelligence generation	Medium	Medium	ML model validation, analyst review

The placement of honeypots raises ethics and legal issues. All experiments presented in this article were performed in isolated cloud environments based on synthetic attack traffic. No actual adversary information was collected, stored or disseminated. The system design includes the following privacy-preserving mechanisms: the anonymisation of source IP addresses in published intelligence feeds, subject to any relevant Data Protection legislation [25].

V. SYSTEM ARCHITECTURE AND DESIGN

5.1 Architectural overview

The proposed system makes use of the microservices architecture deployed on the Kubernetes, which consists of four main layers: (i) the deception layer, (ii) the data ingestion layer, (iii) the analysis and classification layer, and (iv) the intelligence dissemination layer.

5.2 Deception layer

The deception layer consists of containerised honeypot services, each of which emulates a protocol that is typically targeted. Supported services are: SSH (based on Cowrie [9]) and HTTP/HTTPS (custom Flask based application) MySQL (custom emulator) Remote Desktop Protocol (RDP) (PyRDP based [26]). Each honeypot service is packaged as an Open Container Initiative (OCI)-compliant container image with configurable levels of fidelity of the interaction. Kubernetes Deployment manifests are created for the number of replicas, the resources limits and liveness probes of each honeypot type. A Horizontal Pod Autoscaler (HPA) is used to dynamically scale the number of replicas of honeypot pods deployed based on observed connection rates, and thus any sudden surge in attack traffic is handled without any service degradation.

5.3 Data ingestion laye

All honeypot services emit structured log events using a side car container that runs Fluentd [27], which sends events to an Apache Kafka message queue [28]. Kafka offers durable and partitioned message storage that decouples the log production and log consumption thereby absorbing bursts of traffic and not losing data. A normalisation microservice

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 8.699

|www.ijirset.com |A Monthly, Peer Reviewed & Refereed Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|

Volume 15, Issue 2, February 2026

|DOI: 10.15680/IJIRSET.2026.1502103|

subscribes to Kafka topics, converts heterogeneous log formats to unified format using a common schema in a standard format (JSON) and writes normalised records to a TimescaleDB time-series database [29].

5.4 Analysis and classification layer

The analysis layer has three components. The machine learning classifier implementation scikit-learn [30] and TensorFlow [31] takes the normalised log records and classifies each session based on the MITRE ATT&CK framework [24]. The model is based on an ensemble of gradient boosted decision trees (XGBoost) trained on a set of labelled honeypot datasets and is based on features derived from session duration, command sequences and payload entropy and network flow characteristics.

The pattern correlation engine finds relationships between classified sessions - for example, coordinated scanning from a common autonomous system number (ASN) or sequential exploitation of multiple services from a single campaign. The IoC extraction module extracts session data and parses it to find file hashes, domain names, IP addresses, URLs and YARA rule matches [32].

5.5 Intelligence dissemination layer

Extracted IoCs and classified attack patterns are serialised into the STIX 2.1 bundles [18] by the STIX generator microservice. These bundles are published to a built-in TAXII 2.1 server [19], to allow their consumption in an automated way by subscribing organisations and SIEM platforms. A web-based dashboard, built with Grafana [33], is provided for real-time visualisation of attack trends, geographic origin distribution and classification breakdowns.

VI. METHODOLOGY

6.1 Research approach

This research relies on a design science methodology [34], which consists of from iterative cycles of artefact design, implementation, evaluation and refinement. The artefact-which is the cloud native honeypot system-is evaluated against quantitative metrics that include detection efficacy, classification accuracy, system performance and resource utilisation.

6.2 Dataset preparation

Training data for the machine learning classifier was put together from three data sources: (i) datasets of honeypot logs made publicly available from the CAIDA repository [35]; (ii) synthetic attack sessions generated using the MITRE Caldera adversary emulation framework [36]; and (iii) benign traffic baselines captured from production cloud workloads (anonymised). The combined dataset consisted of 147,832 labelled sessions in 14 categories of ATT&CK techniques.

Feature engineering resulted in 63 features per session, such as statistical summaries of inter-arrival times, n-gram frequency of shell commands, Shannon entropy of transferred payloads and categorical encodings of protocol-specific fields.

6.3 Model training and validation

The XGBoost classifier was trained with five-fold stratified cross validation with hyperparameter optimisation using Bayesian search (Optuna framework [37]). The objective function maximised the macro-averaged F1-score to take care of the class imbalance effect. The best hyperparameters gave a max tree depth of 8, learning rate of 0.05, 500 estimators and minimum child weight of 3.

6.4 Deployment environment

The system has been deployed in a managed Kubernetes cluster (Google Kubernetes Engine, GKE) consisting of three pools of nodes: (i) a deception pool which has four n1-standard-2 instances; (ii) an analytics pool which has two n1-standard-4 instances; and (iii) a management pool which has one n1-standard-2 instance. Network policies were used to provide strict micro-segmentation between honeypot pods and backend services. Egress out of honeypot pods was limited to DNS resolution and log endpoint connections only to prevent this from being used as a pivot point.

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 8.699

|www.ijirset.com |A Monthly, Peer Reviewed & Refereed Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|

Volume 15, Issue 2, February 2026

|DOI: 10.15680/IJIRSET.2026.1502103|

6.5 Experimental design
Three experimental scenarios were developed:

6.5.1 Scenario A - Automated scan detection Nmap [38] and Masscan were set to perform full port scan and service enumeration against the honeypot deployment from external virtual machines.

6.5.2 Scenario B - Targeted exploitation:*The Metasploit Framework [39] was used to carry out multi-stage attacks, which comprised SSH brute forcing, web application exploitation (SQL injection, command injection) and SMB exploit delivery.

6.5.3 Scenario C - APT emulation: MITRE Caldera [36] was set up with adversary profiles for the APT29 and APT3 threat groups, running technique chains which include initial access, credential dumping, lateral movement and data exfiltration.

VII. IMPLEMENTATION AND TESTING

7.1 Containerisation and deployment
Each honeypot service was containerised using multi-stage Docker builds to minimise image size and reduce the attack surface of the container itself. Table summarises the container image specifications.

Service	Base Image	Image Size (MB)	Exposed Ports
SSH	python:3.11-slim	142	22, 2222
HTTP	python:3.11-slim	138	80, 443, 8080
SMB	debian:bookworm-slim	167	445
MySQL	python:3.11-slim	145	3306
RDP	python:3.11-slim	189	3389

Helm charts were developed to parameterise deployments, enabling single-command installation with configurable replica counts, resource quotas and service exposure modes (NodePort, LoadBalancer or Ingress).

7.2 Unit and integration testing
A comprehensive test suite comprising 312 unit tests and 48 integration tests was developed using pytest. Unit tests validated individual component behaviours—log parsing accuracy, feature extraction correctness and STIX bundle schema compliance. Integration tests verified end-to-end data flow from simulated connection through to intelligence output. Code coverage exceeded 87% across all modules.

7.3 Security hardening
Several security hardening measures were implemented: (i) all containers were executed as non-root users with read-only root filesystems; (ii) Kubernetes NetworkPolicies enforced deny-all-ingress defaults with explicit allow rules; (iii) Pod Security Standards were applied at the namespace level; (iv) container images were scanned for vulnerabilities using Trivy [41] as part of the continuous integration pipeline; and (v) secrets were managed via HashiCorp Vault [42] with automatic rotation.

VIII. RESULTS AND DISCUSSION

8.1 Detection efficacy
Table shows the results of the detection in the three experimental scenarios.
Detection efficacy across experimental scenarios

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 8.699

|www.ijirset.com |A Monthly, Peer Reviewed & Refereed Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|

Volume 15, Issue 2, February 2026

|DOI: 10.15680/IJIRSET.2026.1502103|

Concurrent Connections	Pod Replicas	Mean Response Latency (ms)	Log Processing Throughput (events/s)
100	3	45	1,200
500	5	52	5,800
1,000	8	61	11,400
5,000	15	89	48,200
10,000	24	134	87,600

The overall detection rate of this system was 97.2% with maximum effectiveness against automated scanning attack (99.0%) and slightly less effective against targeted and APT emulated attacks (93.6% and 94.4%, respectively). The undetected instances in Scenarios B and C were due to encrypted tunnelling techniques, which evaded protocol-level inspection, and highly evasive command sequences which were below the confidence level of the classifier.

8.2 Classification accuracy

The XGBoost classifier has achieved a macro-averaged F1-score of 0.912 on the held-out test set: performance for each class is shown in the Table.

Table for Classification performance by ATT&CK tactic category

ATT&CK Tactic	Precision	Recall	F1-Score	Support
Reconnaissance	0.94	0.96	0.95	4,218
Initial Access	0.92	0.90	0.91	3,847
Execution	0.89	0.91	0.90	2,956
Persistence	0.88	0.85	0.86	1,432
Privilege Escalation	0.90	0.87	0.88	1,105
Credential Access	0.93	0.94	0.93	2,673
Lateral Movement	0.87	0.83	0.85	894
Exfiltration	0.91	0.89	0.90	1,241

The highest F1-scores were scored for Reconnaissance (0.95) and Credential Access (0.93), which are signatures that are distinctive and well represented for scanning and brute-force activities. Lateral Movement had the lowest F1-score (0.85), due to the subtlety and variety of techniques of inter-host traversal.

8.3 Threat intelligence output

During the 30-day evaluation period, the system identified 2,847 STIX bundles comprised of 12,394 indicators (IP addresses, file hashes, domain names), 847 attack patterns and 156 campaign objects. Automated validation against the VirusTotal and AbuseIPDB threat intelligence services found 78.3 percent of extracted IoC based on IP to be independently corroborated as malicious, strong evidence of the intelligence quality of the system.

IX. EXPERIMENTAL EVALUATION AND PERFORMANCE TESTING

9.1 Resource utilisation

Resource consumption was monitored with Prometheus [43] and Grafana at the 30-day evaluation time. Table summarises the mean resource utilisation per component.

Mean resource utilisation of each system component

Component	CPU (millicores)	Memory (MiB)	Network In (Mbps)	Network Out (Mbps)
SSH Honeypot (per pod)	85	128	2.1	0.4
HTTP Honeypot (per pod)	112	156	4.7	1.2

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 8.699

|www.ijirset.com |A Monthly, Peer Reviewed & Refereed Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|

Volume 15, Issue 2, February 2026

|DOI: 10.15680/IJIRSET.2026.1502103|

Kafka Broker	340	1,024	8.3	6.1
ML Classifier	520	2,048	1.2	0.3
STIX Generator	95	256	0.4	0.8
Total Cluster	2,840	8,192	–	–

The total cluster resource consumption was far below the provisioned capacity with the CPU utilisation averaging 34% and memory utilisation averaging 52% under sustained attack traffic. These figures prove the efficiency of the system in terms of its resource use.

9.2 Scalability assessment

Horizontal scaling behaviour was assessed by stepping up from 100 to 10,000 simulated attack intensity of concurrent connections. Table explains the relationship between connection volume and key performance metrics.

Concurrent Connections	Pod Replicas	Mean Response Latency (ms)	Log Processing Throughput (events/s)
100	3	45	1,200
500	5	52	5,800
1,000	8	61	11,400
5,000	15	89	48,200
10,000	24	134	87,600

Scalability metrics which display pod replica count, response latency and log throughput as functions of concurrent connection volume

The HPA successfully scaled honeypot pods proportionally to load with latency increasing sub-linearly from 45 ms to 134 ms with a 100-fold increase in concurrent connections. Log processing throughput increased almost linearly, which proved the effectiveness of the Kafka partitioned architecture.

X. CASE STUDY: SIMULATED SECURITY INCIDENT

10.1 Scenario description

A complete simulated security setup was carried out to ensure the end-to-end system pipeline. The scenario modelled a multi-stage attack by a hypothetical threat actor ("CRIMSON FALCON") on an organisation's web application infrastructure on the cloud.

10.2 Attack execution

The attack consisted of 5 phases: (i) external reconnaissance through port scanning (Nmap) and web application fingerprinting (WhatWeb); (ii) initial access through SQL injection against the honeypot in the SSH server; (iii) credential harvesting through simulated database credential extraction; (iv) lateral movement through harvested credentials to authenticate to the honeypot in the SSH server; and (v) simulated data exfiltration through encrypted HTTPS channels.

10.3 System response

The system recognized and reacted to each phase of attacks as follows. The reconnaissance stage was detected within 1.1 seconds, generating an automated warning as well as triggering enhanced logging. The malware's attempt to insert its SQL into the database was labeled as Initial Access (T1190 - Exploit Public-Facing Application) at 0.94 confidence. The Credential extraction was mapped to Credential Access (T1552 - Unsecured Credentials) The SSH lateral movement was correlated with the earlier occurring HTTP session based on temporal proximity and shared source ASN which produced a unified campaign object. The attempt at exfiltration was detected by the anomaly detection module based on abnormal patterns of the volume of data being sent out.

10.4 Intelligence output

Within 4.7 minutes of the initial reconnaissance the system received ready a complete STIX 2.1 bundle containing 1 Threat Actor object, 1 Campaign object, 5 Attack Pattern objects (mapped to ATT&CK technique) 8 Indicator objects

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 8.699

|www.ijirset.com |A Monthly, Peer Reviewed & Refereed Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|

Volume 15, Issue 2, February 2026

|DOI: 10.15680/IJIRSET.2026.1502103|

(IP addresses, SQL injection payloads, SSH session identifiers) and 3 Relationship objects between campaign elements. This bundle was automatically published to the TAXII server and was a consumable bundle by subscribing SIEM platforms within seconds.

XI. LIMITATIONS AND CONSIDERATIONS

A number of limitations must be noted. Firstly, the evaluation used simulated traffic, rather than real-world, for the adversary. Whilst the attack emulation frameworks used (Metasploit, Caldera) are generally accepted as being representative, real adversaries may be characterized by behaviours not modelled within these tools [45].

Secondly, the performance of the machine learning classifier is dependent on the representativeness and quality of the training data. Periodic model retraining should be employed because of the fact that the tactics of adversaries change over time which may cause degradation of classification accuracy.

Thirdly, currently the system supports five network protocols. Production deployments would benefit from wider range of protocols coverage, which includes Internet of Things (IoT) protocols (MQTT, CoAP) and industrial control system (ICS) protocol (Modbus, DNP3).

Fourthly, sophisticated adversaries might use honeypot detection techniques, such as timing analysis, service fingerprinting and deliberate interaction testing. Whilst the system includes emulation of real services, a determined Tier 3 adversary may be able to see through the deception infrastructure.

Finally, the ethical and legal aspects of honeypot deployment differ from jurisdiction to jurisdiction. Organisations need to ensure that they respect related legislation on data capture, storage and sharing, especially on relevant information that can be used to identify individuals - the personally identifiable information within attack traffic [25].

XII. CONCLUSION

This paper presented the design, implementation and evaluation of a system of honeypot creation tool for the purpose of an automated threat intelligence generation system, which is cloud native. The system uses orchestration provided by Kubernetes, containerised honeypot services, attack classification based on machine learning and standardised intelligence formats (STIX/TAXII) to achieve a provision of scalable, automated and operationally efficient deception infrastructure.

Experimental evaluation under three types of attack scenarios showed a total overall detection rate of 97.2% and mean latency of 2.1 seconds. The XGBoost classifier obtained a macro-averaged F1-score of 0.912 for 8 ATT&CK tactics categories. In near-linear log processing throughput under 100-fold increases on concurrent connections, testing for all increased scales of concurrent connections were determined to be sub-linear latency degradation. The simulation security incident case study confirmed the system's ability to detect, classify, correlate and disseminate the threat intelligence within minutes of the initial compromise.

The main contributions of this work are threefold: (i) the first integrated cloud native honeypot architecture using automatic STIX/TAXII intelligence generation, (ii) the use of gradient enhanced ensemble classifiers directly in the honeypot data pipeline for real-time ATT&CK tactic classification and (iii) extensive empirical evaluation under realistic cloud deployment conditions using quantified performance benchmarks.

Cloud-native honeypots are a major breakthrough in proactive cyber defence, enabling deception infrastructure to move beyond passive monitoring infrastructure to intelligence-generating active infrastructures that can scale with the dynamic nature of a cloud environment.

XIII. FUTURE ENHANCEMENTS

A number of directions for future work are identified. Firstly, the combining large language models (LLMs) for natural language threat intelligence reporting from structured STIX data would further reduce analyst workload. Secondly,

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 8.699

|www.ijirset.com |A Monthly, Peer Reviewed & Refereed Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|

Volume 15, Issue 2, February 2026

|DOI: 10.15680/IJIRSET.2026.1502103|

federated learning approaches could be used to make collaborative learning possible across multiple organisations without federated learning sharing sensitive honeypot data. Thirdly, in extending the deception layer to cover serverless function honeypots (e.g. AWS Lambda), cloud storage bucket honeytokens and container registry poisoning detectors would provide wider coverage. Fourthly, adversarial machine learning techniques could be explored to improve classifier robustness to the evasion attempts. Finally, integration with software-defined networking (SDN) controllers would allow for dynamic traffic redirection to honeypots based upon real-time risk scoring to create an adaptive deception mesh.

XIV. ACKNOWLEDGEMENTS

The authors express their sincere thanks to the Department of Computer Science & Engineering,Parul University for their continuous guidance and sincere support throughout this research work. Special thanks are also extended to our esteemed project mentor Akruti Pandwal and the faculty coordinators for their invaluable feedback, insightful suggestions and constant encouragement throughout the whole development & evaluation phase of the Cloud- Native Honeypot For Automated Threat.

REFERENCES

[1] Cloud Native Computing Foundation, “Cloud Native Definition v1.0,” 2024. Available: https://github.com/cncf/toc/blob/main/DEFINITION.md
[2] Kubernetes Project, “Kubernetes Documentation,” 2025. Available: https://kubernetes.io/docs/home/
[3] OpenTelemetry Project, “OpenTelemetry Documentation,” 2025. Available: https://opentelemetry.io/docs/
[4] Prometheus Authors, “Prometheus Monitoring Documentation,”2025. Available:https://prometheus.io/docs/introduction/overview/
[5] Grafana Labs, “Grafana Documentation,” 2025. Available: https://grafana.com/docs/
[6] OASIS. STIX Version 2.1 Standard. Available: https://docs.oasis-open.org/cti/stix/v2.1/stix-v2.1.html
[7] Honeynet Project, “Know Your Enemy: Honeynet Project Research,” 2024. Available: https://www.honeynet.org/
[8] MITRE Corporation, “ATT&CK Knowledge Base,” 2025. Available: https://attack.mitre.org/
[9] DARPA, “Intrusion Detection Evaluation Datasets,” 2024. Available: https://www.ll.mit.edu/r-d/datasets
[10] CAIDA, “Internet Security and Attack Datasets,” 2024. Available:https://www.caida.org/catalog/datasets/
[11] Canadian Institute for Cybersecurity, “CICIDS2017 Dataset,” 2024. Available: https://www.unb.ca/cic/datasets/ids-2017.html
[12] OASIS Open, “STIX Version 2.1 Standard,” 2021. Available: https://docs.oasis-open.org/cti/stix/v2.1/stix-v2.1.html
[13] FIRST.Org, “Traffic Light Protocol (TLP),” 2024. Available: https://www.first.org/tlp/
[14] CISA, “Cybersecurity Publications and Threat Intelligence Resources,” 2025. Available: https://www.cisa.gov/publications
[15] ENISA, “ENISA Threat Landscape Reports,” 2025. Available: https://www.enisa.europa.eu/publications
[16] NIST, “Guide to Intrusion Detection and Prevention Systems (SP 800-94),” 2007. Available: https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-94.pdf
[17] UK National Cyber Security Centre, “Cyber Security Guidance,” 2025. Available: https://www.ncsc.gov.uk/guidance
[18] OWASP Foundation, “OWASP Documentation,” 2025. Available: https://owasp.org/
[19] Internet Engineering Task Force, “Security RFC Standards,” 2025. Available: https://www.ietf.org/standards/rfcs/
[20] arXiv, “Open Access Cybersecurity Research Papers,” 2025. Available: https://arxiv.org

INTERNATIONAL STANDARD SERIAL NUMBER INDIA

INNO SPACE SJIF Scientific Journal Impact Factor

doi cross ref

निस्केयर NISCAIR

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN SCIENCE | ENGINEERING | TECHNOLOGY

9940 572 462 6381 907 438 ijirset@gmail.com

www.ijirset.com

Scan to save the contact details