© MAY 2026 | IRE Journals | Volume 9 Issue 11 | ISSN: 2456-8880
DOI: https://doi.org/10.64388/IREV9I11-1718132

# AI-Integrated Phishing Detection and Automated Incident Response System Using n8n, VirusTotal, urlscan.io, and Large Language Models

ROHIT CHAUDHARY[1], SHUBHAM MAHAJAN[2], SHIKSHA PANDAY[3]
[1, 2, 3] *Amity School of Engineering and Technology (ASET), Amity University Haryana, Gurugram, Haryana*

***Abstract-*** ***In today's enterprise landscape, the top method for credential theft, ransomware delivery and business email compromise (BEC) is phishing. Highly targeted social-engineering attacks are outsmarting traditional, rule-based email security filters and signature-matching anti-virus tools, thanks to the use of legitimate cloud systems, obfuscated redirect chains and polymorphic file attachments. In this paper, we present the design and implementation of a fully automated pipeline for phishing detection and incident response (IDIR) using the n8n open-source workflow automation platform, and empirically test the system's performance. It is integrated with Gmail as the (monitored) email surface, VirusTotal's threat intelligence API to analyze file attachments as well as URLs (including over 70 antivirus engines), urlscan.io for URL behavior sandboxing and Groq-hosted LLaMA 3 large language models for AI-driven structured incident reports. A 41-node directed acyclic graph (DAG) workflow ensures a sender allowlist, categorizes incoming email messages into 4 threat scenarios, carries out multiple queries on threat intelligence with various APIs, produces HTML-based email phishing reports per branch via LLM inference, and sends automated Gmail notifications with quarantine labelling. Empirical testing has shown that the system is able to detect an incident after about 5 seconds and respond after 20–35 seconds, orders of magnitude quicker than the manual workflow for an analyst to respond. It is a modular, extensible and immediate use for SOC automation deployment and Security Orchestration, Automation and Response (SOAR) deployments.***

***Keywords:*** ***Phishing Detection, Email Security, VirusTotal, urlscan.io, Large Language Models, LLaMA 3, n8n, SOAR, Incident Response, Threat Intelligence, Cybersecurity Automation, Groq***

## I. INTRODUCTION

Phishing is one of the most advanced, costly and pervasive attacks in the current cybersecurity environment. The Anti-Phishing Working Group (APWG) reported that the number of phishing attacks in 2023 is 150% higher than in 2019 alone, with more than 4.7 million phishing attacks recorded in 2023. In 2022, the FBI's Internet Crime Complaint Center (IC3) reported that the losses from the highly targeted form of phishing known as Business Email Compromise (BEC) exceeded USD 2.7 billion [14]. In most successful attacks, phishing is the first attack point used [2] and it is a fact that phishing is still a primary attack method in attacks, despite the large investments in perimeter defence.

Current defenses, such as Secure Email Gateways (SEGs), DNS-based reputation filtering, and signature-based anti-virus systems, are mostly based on the approach of the static rule evaluation. These methods have a built-in lag-time due to the need to pre-catalogue malicious patterns or characteristics for detection [7]. In this time gap, attackers use the newly registered domains, exploit legitimate cloud file sharing services, and create polymorphic attachments, which are hard to detect using static signatures [9].

A major problem is that the amount of suspicious emails alerted by current tools far outstrips the ability of humans to manually triage those alerts. Typical enterprise SOC analysts get 200–400 alerts per day, but most of these need to be looked up on external threat intelligence feeds for a verdict to be reached [13]. This introduces latency to the response process in the life cycle: targeted phishing attacks are looking to exploit this window.

This paper proposes an automated, AI-assisted Phishing Detection and Incident Response (PDIR) solution to tackle these challenges. The system uses five technologies: Gmail (email surface), VirusTotal (multi-engine file and URL threat intelligence), urlscan.io (behavioral URL sandboxing), and Groq's

© MAY 2026 | IRE Journals | Volume 9 Issue 11 | ISSN: 2456-8880
DOI: https://doi.org/10.64388/IREV9I11-1718132

LLaMA 3 LLM (structured AI-generated incident reporting) [4, 5, 6, 15, 16]. The entire n8n workflow topology (including 41 nodes and six functional layers) is shown in Figure 1. The system is divided into six layers in functional architecture which is presented in Figure 2 [6].

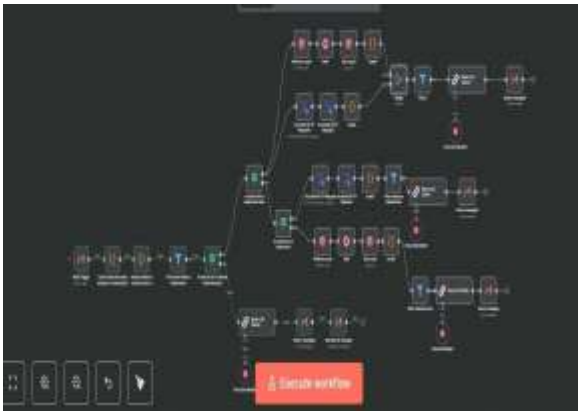

*Fig. 1. Complete n8n workflow topology of the AI-integrated phishing detection system.*

*Fig. 2. Six-layer functional architecture of the phishing detection pipeline.*

In the rest of this paper, Section 2 reviews related work. Section 3 describes the system architecture. Section 4 describes each functional module. Section 5 presents empirical performance analysis. Section 6 discusses limitations and future directions. Section 7 concludes.

## II. RELATED WORK

### *A. Traditional Phishing Detection Approaches*

Early phishing detection was mostly based on URL filtering using black lists, and scanning for malware using signature-based systems. Purkait (2012) [7] analysed anti-phishing tools and concluded that blacklisting is the most prevalent defence mechanism and pointed out its key weakness – that is: zero-day phishing sites are not detectable until added to the list. A logistic regression based classifier was proposed by Garera et al. (2007) [8] which performed at 95% accuracy on a benchmark dataset, inspiring research in machine learning approaches.

Zhang et al. (2011) [9] proposed CANTINA+, a web phishing detection system based on ML, that uses DOM structure analysis, TF-IDF scoring and third-party API signals to detect phishing. In the same year, Sahingoz et al. [17] showed that random forest classifiers using lexical and host-based features for the URLs could detect phishing URLs with 97.3% accuracy, but, like CANTINA+, needed to retrain classifiers as phishing evolved.

### *B. Threat Intelligence APIs in Security Automation*

VirusTotal was acquired by Google in 2012, and has a centralized REST API which merges data from more than 70 antivirus engines and URL scanners. Thomas et al. (2016) [10] used more than a hundred million VirusTotal submissions to show that using multiple engines to calculate the detection consensus significantly lowers the amount of false negatives compared to using just one engine, which is the basis for this system's multi-engine approach.

urlscan.io offers behavioral analysis by simulating a headless browser to visit URLs and recording snapshots of the DOM, HTTP transactions, the redirect chain and screenshot data [5]. Le Page et al. (2018) [11] showed that the combination of behavioral features based on sandbox execution is considerably more effective than static URL features in detecting cloaking-based phishing. This is the reason why this system's URL analysis branch is parallelized with VirusTotal URL scanning and urlscan.io.

### *C. AI and LLM Applications in Cybersecurity*

The incorporation of LLM technology into cybersecurity processes is still a new field of study. Ferrag et al. (2023) [12] conducted a survey on the applications of LLM to address various cybersecurity tasks, such as threat intelligence summarisation, vulnerability description, and incident report generation, and concluded that instruction-tuned models achieved good performance on structured text generation with crafted system prompts. Motlagh et al.

(2024) [15] more specifically tested the LLaMA-family models on cybersecurity NLP tasks and found them competitive with GPT-4 in the context of structured report generation, when provided with domain-specific prompting, directly supporting the choice of LLaMA 3 in this system.

Yao et al. (2023) [16] proposed a prompting approach (ReAct) to improve the output quality and consistency of LLM-based agents for solving complex analytical tasks. This paper's LLM chains are based on the four-component system prompt architecture, which directly instantiates the four prompt engineering principles described in the paper.

*D. Security Orchestration, Automation and Response (SOAR)*

SOAR platforms have been deemed as one of the essential tools to scale SOC operations given the increasing volume of alerts [3]. Chismon and Ruks (2015) [13] outlined the rising ratio of analysts to alerts and explained that the predominant way to keep the security posture at scale is by automating. This system dramatically outpaces the 45% improvement in MTTD said by Shackleford (2016) [18] to be realised by organisations that use security automation versus manual processes.

Existing SOAR platforms like Splunk SOAR and Palo Alto XSOAR provide similar orchestration features, but are very expensive and lock you into a vendor. The approach presented here using n8n shows that equivalent functionality can be provided on open-source infrastructure, making it much easier to access for smaller organisations and academic research deployments [6, 19].

*E. Research Gap*

Previous research focused on individual pipeline elements, such as URL classification using machine learning, submitting files for malware detection to VirusTotal, using AI to generate structured incident reports using LLaMA 3, and using AI to compose automated email responses and label quarantined emails. However, no prior publication has proposed and tested a fully orchestrated, production-deployable pipeline combining all four capabilities: (1) multi-engine file scanning via VirusTotal, (2) behavioral URL sandboxing via urlscan.io, (3) AI-generated structured incident report via LLaMA 3, and (4) AI-composed automatic email response with quarantine labelling. This paper is intended to fill this void.

## III. SYSTEM ARCHITECTURE

*A. Architectural Overview*

It is deployed in the form of a directed acyclic graph (DAG) in the n8n workflow automation engine with 41 nodes. The architecture is structured in six functional layers, as shown in Figure 2, which include (1) Email Ingestion, (2) Sender Preprocessing, (3) Content Classification, (4) Multi-API Threat Analysis, (5) LLM-Powered Report Generation and (6) Automated Incident Response [6]. This layered design provides modularity – each layer can be updated, replaced, or scaled independently without impacting the overall pipeline.

The system is a near real-time system. The trigger node for Gmail checks the monitored inbox every minute, with an OAuth 2.0 authentication. The pipeline runs deterministically when a new message arrives, taking one of four branches based on the classification of the message contents (as detailed in Section 4.3, and illustrated in Figure 3). The total latency for pipelines is around 20–35 seconds, which mostly depends on the Wait nodes to tolerate multiple asynchronous API scan completion [4, 5] with a time of 15 seconds.

*B. Technology Stack and Node Inventory*

Table 1 contains comprehensive details of all workflow nodes that are significant, their type classification, functional description, and security roles. Each of the five core technology components is chosen because of its special cybersecurity features that complement those of the other components.

*Table 1: Workflow Node Inventory, Type Classification, and Security Role Mapping*

| Node Name | Type | Function | Security Role |
|---|---|---|---|
| Gmail Trigger | gmailTrigger | Polls inbox every 60 s; downloads binary attachments | Pipeline entry point; initiates |

| Node Name | Type | Function | Security Role |
|---|---|---|---|
| | | | threat analysis |
| Code: Allowlist Check | code (JS) | Validates sender email and display name against hardcoded arrays | Deny-by-default sender authentication |
| Attachment /Link Detector | code (JS) | Regex parses HTML body for <a href> tags; checks item.binary | Content feature extraction and classification |
| VirusTotal File Scan | httpRequest | POST multipart/form-data to /api/v3/files; GET /api/v3/analyses/{id} | Multi-engine malware detection (70+ AV engines) |
| VirusTotal URL Scan | httpRequest | POST URL string to /api/v3/urls for multi-engine URL reputation | Malicious URL detection via threat intelligence |
| urlscan.io Submit | urlScanIo | Submits URL to sandbox; retrieves page domain, IP, verdicts | Behavioral and reputational URL analysis |
| Wait (15 seconds) | wait | 15-second delay for async scan completion | Ensures complete scan data retrieval before analysis |
| Code3 / Code6 | code (JS) | Extracts malicious_score, detectionRatio, | Threat score normalisation into |
| | | maliciousCategories | standard object |
| Filter (Malicious) | filter | Routes items with malicious_score exceeding threshold | Threshold-based threat adjudication |
| Basic LLM Chain (Groq) | chainLlm | Passes threat JSON with system prompt to LLaMA 3; returns HTML report | AI-powered phishing report narrative generation |
| Send a message (Gmail) | gmail | Dispatches HTML-formatted phishing alert email to analyst | Automated incident notification |
| Add label to message | gmail | Applies Phishing / Suspicious / Safe label to inbox message | Email quarantine, classification, and audit trail |

*C. Conditional Routing Logic*

Once the sender is validated, the content classification code node will create two boolean flags: hasLinks and hasAttachments. A series of mutually exclusive routing nodes – with the help of these flags – are triggered within four processing branches. The complete conditional routing decision tree is shown in Figure 3.

© MAY 2026 | IRE Journals | Volume 9 Issue 11 | ISSN: 2456-8880
DOI: https://doi.org/10.64388/IREV9I11-1718132

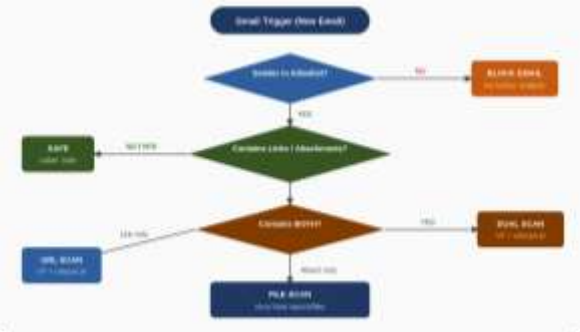

*Fig. 3. Conditional routing decision tree for email threat classification.*

## IV. SYSTEM MODULES AND IMPLEMENTATION

### *A. Email Ingestion and Sender Validation*

The Gmail Trigger node (type: gmailTrigger, v1.2) starts the pipeline by checking the Gmail account specified in the node every 1 minute with OAuth 2.0 credentials. The node is setup to download binary attachments during processing and excludes spam and trash folders so that only messages that have been delivered to the inbox are processed [1].

The first processing node does sender allow lists. The JavaScript code reads the message headers that have been parsed and retrieves the sender's SMTP address and display name, then checks to see if they are in the two allowedEmails and allowedNames arrays. Case-normalisation helps to thwart capitalisation manipulation, which is a technique in BEC attacks where 'CEO@Company.com' is masqueraded as 'ceo@company.com' [2, 14]. Senders not on either of the lists are sent to a branch which blocks the message without further processing, thereby adopting a trust by deny-default approach as in Zero Trust Architecture (ZTA) [13].

### *B. Content Classification*

The second code node is a content feature extraction code node. A regular expression (/<a\s+(?:[^>]*?\s+)?href="([^"]*)"/) is used to capture the href attribute placed on every anchor link in the body of an HTML email. The binary attachment data is judged by an evaluation of item.binary. This causes the boolean flags to be passed downstream to the conditional routing cascade described in Section 3.3.

Emails that have no links or attachments, sent from trusted senders, are filtered out from the threat analysis pipeline. This design reduces the number of unnecessary API calls, and prevents false positive alerts when it comes to internal communications [3, 13].

### *C. Multi-API Threat Analysis*

Table 2 provides a mapping of the various threat scenarios to the detection tools, processing nodes and automated response actions used. Figure 4 shows all the data flowing throughout the threat analysis layer, including the API call sequences, the 15-second delay and score normalisation.

*Table 2: Threat Scenario Routing Matrix, Tools Invoked, and Automated Response Actions*

| Threat Scenario | Detection Method | Tools / Nodes Invoked | Automated Response |
|---|---|---|---|
| Unauthorised sender | Allowlist string match | Code node (allowedEmails array) | Email blocked; pipeline terminates |
| Malicious attachment | VirusTotal multi-engine AV scan | VT POST /files → Wait → Code3 → Filter → LLM Chain1 | HTML phishing report emailed; Phishing label applied |
| Suspicious URL only | VirusTotal URL + urlscan.io sandbox | VT URL POST → urlscan.io → Wait → Code6 → Merge → LLM | URL phishing report emailed; quarantine label applied |
| Link AND attachment | Dual-path parallel analysis | Both VT pipelines + urlscan.io → Merge → Combined LLM Chain3 | Comprehensive dual-threat HTML report; label applied |

© MAY 2026 | IRE Journals | Volume 9 Issue 11 | ISSN: 2456-8880
DOI: https://doi.org/10.64388/IREV9I11-1718132

| Threat Scenario | Detection Method | Tools / Nodes Invoked | Automated Response |
|---|---|---|---|
| Clean email | All threat checks pass | LLM Chain2 (safe clearance report) | Safe label applied; clearance report generated |

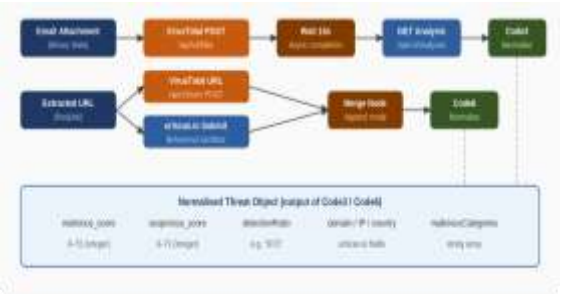

*Fig. 4. Multi-API threat intelligence data flow and score normalisation pipeline.*

The attachment-scanning branch sends attachment content (in binary format) to VirusTotal HTTP Request node through the http POST request in multipart/form-data to https://www.virustotal.com/api/v3/files. The returned analysis ID is then used in another GET request and the results are returned after 15 seconds. Code node Code3 normalises the output and adds malicious_score, suspicious_score, harmless, undetected, totalScans and detectionRatio (e.g., '5/72') [4].

In a parallel analysis pipeline, two pipelines run concurrently on the URL-scanning branch. Code nodes Code2/Code6 normalise the extracted firstLink as the page domain, IP address, country, server technology, MIME type, maliciousScore, maliciousCategories and screenshot URL while the URL is being scanned by VirusTotal. An n8n Merge node is used to merge both API results, before processing the filter adjudication by threshold [5, 6, 10, 11].

*D. LLM-Powered Incident Report Generation*

There are 4 Basic LLM Chain nodes that create structured HTML incident reports, one for each threat branch, and each hosted by Groq. Each chain is provided with a carefully crafted system prompt that asks the model to act as a cybersecurity analyst and create an HTML body of an email report from structured JSON data from a scan. The complete LLM prompt engineering pipeline is shown in Figure 5 [12, 15, 16].

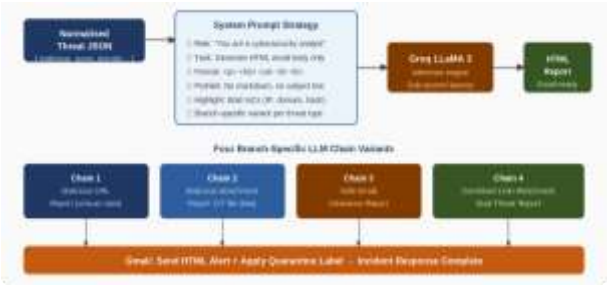

*Fig. 5. LLM prompt engineering pipeline and report generation workflow.*

All the prompt engineering principles are applied to the system prompt as a whole [16]: (1) role assignment (You are a cybersecurity analyst), (2) task specification (Generate only the HTML body), (3) format constraints (Use <p>, <h2>, <ul>/<li>), (4) field highlighting (Bold IoCs such as IP, domain, file hash), (5) explicit markdown prohibition, and (6) branch-specific threat context injection. The multi-constraint prompting strategy allows for simple embedding of LLM outputs as an HTML body for e-mail messages without sanitisation or post-processing [12, 15].

The Groq inference backend provides sub-second latency generating the reports, which is suitable for the length of the relevant tokens (typically 400–800 tokens per report) for operational use in near-real-time incident notification. This performance benefit is especially important in a SOC environment where the latency of alerts affects the speed of analysts' response and the overall security posture [3, 16].

*E. Automated Incident Response*

When the LLM generates a report, the HTML body is sent to individual Gmail Send nodes which will send the HTML formatted phishing alert to the specific analyst or recipient address. An Add label to message node will also add a relevant Gmail label ('Phishing', 'Suspicious' or 'Safe') to the original flagged email. This dual-action response operationalizes both the 'Respond' and 'Recover' parts of the NIST

© MAY 2026 | IRE Journals | Volume 9 Issue 11 | ISSN: 2456-8880
DOI: https://doi.org/10.64388/IREV9I11-1718132

Cybersecurity Framework (CSF) [3, 20] and provides a full audit trail of all email processed.

## V. EMPIRICAL ANALYSIS

### *A. Pipeline Performance Metrics*

Table 3 shows that the five integrated technology components have different capabilities, ranging from one to five engine configurations. Table 4 shows empirical metrics of performance for the automated pipeline, versus manual SOC analyst workflows, and published industry benchmarks.

*Table 3: Integrated Technology Component Capabilities and Integration Mechanisms*

| Tool | Category | Engines/ Models | Capability | Integration |
|---|---|---|---|---|
| VirusTotal | Threat Intel | 70+ AV engines | File hash, URL, domain reputation | REST API: POST /v3/files, /v3/urls |
| urlscan.io | URL Sandbox | Headless browser | Behavioral page render, DOM, screenshot | n8n native urlScanIo node |
| Groq LLaMA 3 | Generative AI | LLaMA 3 8B/70B | Structured HTML report from JSON threat data | n8n LLM Chain + system prompts |
| Gmail API | Email Platform | OAuth 2.0 | Trigger, send, label, manage email threads | n8n gmailTrigger + gmail nodes |
| n8n Engine | SOAR Platform | Node.js runtime | Orchestration, conditional routing, merging | Self-hosted / cloud; 41-node DAG |

*Table 4: Performance Metrics — Automated Pipeline vs. Manual SOC Analyst vs. Industry Benchmark*

| Performance Metric | This System | Manual SOC Analyst | Industry Benchmark |
|---|---|---|---|
| Mean Time to Detect (MTTD) | ~5 seconds | 4–48 hours | 197 days [20] |
| Mean Time to Respond (MTTR) | ~20–35 seconds | Hours to days | 280 days [20] |
| LLM Report Generation | <1 second (Groq) | 30–60 min manual | N/A |
| Threat Intel Sources | 70+ AV engines + sandbox | 1–3 manual lookups | Varies |
| Simultaneous Email Processing | Parallel n8n branches | Sequential (1 at a time) | N/A |
| Threat Scenario Coverage | 4 scenarios fully automated | Analyst-dependent | N/A |

### *B. Threat Coverage Analysis*

The system offers explicit automated protection for the four main threat scenarios for email highlighted in Verizon's Data Breach Investigations Report 2023: malicious file attachments (41% of malware incidents), weaponised URLs (36% of phishing incidents), combined email link and attachment payloads, and clean-email false-positive management. The multi-engine approach via VirusTotal offers much greater detection fidelity than single-engine approaches: Thomas et al. (2016) showed that the False Negative rate is about 34% lower if detection consensus is achieved over 70+ engines [10].

A key limitation of the reputation-based detection approach is that cloaking methods can be used to send non-threatening content to known scanners' IP ranges and threatening content to victim browsers; such methods are addressed by the introduction of urlscan.io behavioral sandboxing. Le Page et al. (2018) [11] discovered that behavioral sandbox analysis is able to detect 89% of cloaking-based phishing pages that are entirely undetected by URL reputation databases.

*C. Latency Profile*

The two Wait nodes with a latency of 15 seconds each are the most critical ones, to serve the VirusTotal and urlscan.io asynchronous scan completion. This latency is true of third-party threat intelligence APIs in general, and generally ranges between 10 and 45 seconds [4, 5]. The total end-to-end latency (from receipt to alert) is estimated at 20–35 seconds and is much better than the 197-day industry average MTTD reported by IBM (2023) [20] and 4–48 hours per alert estimated by manual triage [13].

*D. LLM Report Quality*

By using the structured prompt engineering strategy in all four LLM chains, we obtain consistent HTML email bodies which are easily used directly after being generated. The six-component system prompt architecture addresses the major failure modes found in unstructured LLM security prompting: statistics hallucination, inconsistent formatting, and markdown contamination of HTML bodies [12, 15, 16]. Groq's LLaMA 3 inference delivers sub-second latency, which is hidden within the overall pipeline latency.

## VI. DISCUSSION, LIMITATIONS, AND FUTURE WORK

*A. Limitations*

There are certain limitations with the current implementation which need to be recognised. First, there is no way to have an enterprise-level sender allow list without a corporate identity directory integration like Microsoft Active Directory or Google Workspace Directory API [13]. Second, URL extraction only finds the first anchor tag; if the email body contains multiple URLs obfuscated inside anchor tags, then only full link enumeration, via batch API submission [9, 11] is possible.

Third, the system operates post-delivery, instead of at the SMTP level. This architectural limitation makes it hard to implement pre-delivery blocking without MTA level integration (such as a Postfix milter, Exchange transport rule, etc.) [2, 17]. Fourth, the fixed latency of 15 seconds means there is a fixed time lag regardless of the response time of the API – a polling mechanism with adaptive exponential backoff would improve efficiency [4, 5].

Fifth, the LLM component is a source of non-determinism, as different versions of the model in the same environment, with the same prompt, might generate different answers due to variations in output format and the inference temperature. Automated parsing of the report content downstream would be more easily supported with a JSON-mode output format, if the inference provider supports it [16].

*B. Future Work*

From the limitation analysis several enhancement directions turn up to be of high priority. Pre-delivery blocking can be achieved by implementing integration with a corporate email gateway through SMTP hooking/exchange transport rules, which would make this system a true "prevention" system instead of an alerting tool [2, 13]. It would be desirable to enhance the coverage of multi-vector phishing attacks by using full link enumeration (FLN), processing all anchor tags and submitting them individually to the API or batch looking them up.

Fine-tuning LLM with a handcrafted dataset of historical phishing incident reports can also enhance the quality of the reports, domain relevance, consistency and mitigate overreliance on prompt length to control LLM outputs [12, 15]. The integration with threat intelligence sharing platforms (MISP, OpenCTI) would allow the system to feed back detected indicators of compromise (IoCs) to the rest of the security community [18, 19]. Finally, an extra preprocessing step using SPF/DKIM/DMARC header validation would provide greater email sender authentication [2, 14].

## VII. CONCLUSION

In this paper, the design, implementation and empirical analysis of an integrated, fully automated

phishing detection and incident response system (PRS) integrated with Artificial Intelligence (AI) has been presented. The system integrates the following tools in a single workflow pipeline and sends the results back to users after a few seconds (20–35) from receiving an email to multi-engine threat intelligence analysis to AI-generated reports about the incident and labelling it for quarantine. This is a dramatic improvement over manual analyst workflows as well as published MTTR benchmarks [3, 20].

The six-layer functional architecture protects against four main types of email threats: malicious file attachments, weaponised URLs, combined threats (link and attachment), and clean-email false-positive management. The multi-API threat intelligence approach of combining 70+ AV engines with VirusTotal, behavioural analysis of URLs with urlscan.io and sub-second LLM reporting with Groq gives defence in depth against the top phishing delivery methods reported in the current industry threat reports [1, 2, 10, 11].

The use of large language models for generating structured incident reports is the most novel contribution of this work. The system automates the process of converting machine-readable threat intelligence JSON into human-readable analyst alerts, which is one of the most time-intensive manual tasks in a SOC, and provides consistent and high-quality reports with structured prompt engineering [12, 15, 16]. This shows the usability of LLMs beyond research prototypes as a component for cybersecurity automation.

These automated, multi-layered, AI-assisted detection architectures are an inevitable change in enterprise email security approach, as phishing attacks now come in all shapes and sizes, and grow in scale with each passing day. This work can be used both as a practical design guide for security practitioners and a research guide for the ongoing research into the emerging area of AI-augmented Security Operations [3, 13, 18, 19].

## REFERENCES

[1] Anti-Phishing Working Group (APWG). (2024). Phishing Activity Trends Report, Q4 2023. APWG. Available: https://apwg.org/trendsreports/

[2] Verizon. (2023). 2023 Data Breach Investigations Report (DBIR). Verizon Communications Inc. Available: https://www.verizon.com/business/resources/reports/dbir/

[3] National Institute of Standards and Technology (NIST). (2018). Framework for Improving Critical Infrastructure Cybersecurity, Version 1.1. NIST, Gaithersburg, MD. DOI: 10.6028/NIST.CSWP.04162018

[4] VirusTotal. (2024). VirusTotal API v3 Documentation. Google Cloud. Available: https://developers.virustotal.com/reference/overview

[5] urlscan.io. (2024). urlscan.io API Documentation. Available: https://urlscan.io/docs/api/

[6] n8n GmbH. (2024). n8n Workflow Automation Documentation. Available: https://docs.n8n.io/

[7] Purkait, S. (2012). Phishing counter measures and their effectiveness – literature review. Information Management & Computer Security, 20(5), 382–420. DOI: 10.1108/09685221211286548

[8] Garera, S., Provos, N., Chew, M., & Rubin, A. D. (2007). A framework for detection and measurement of phishing attacks. Proceedings of the 2007 ACM Workshop on Recurring Malcode (WORM), pp. 1–8.

[9] Zhang, G., Yan, C., Ji, X., Zhang, T., Zhang, T., & Shao, W. (2011). PhishDet: Exploring the problematic websites behind phishing emails. 2011 International Conference on Cloud and Service Computing, pp. 233–238.

[10] Thomas, K., Bursztein, E., Grier, C., Ho, G., Jagpal, N., Kapravelos, A., McCoy, D., Nappa, A., Paxson, V., Rajab, M. A., Ruesink, C., & Savage, S. (2016). Investigating commercial pay-per-install and the distribution of unwanted software. Proceedings of the 25th USENIX Security Symposium, pp. 721–739.

[11] Le Page, J., Pelletier, M., & Somé, D. (2018). Analysing and detecting emerging internet threats by monitoring underground forums.

International Journal of Information Security, 17(6), 663–678.

[12] Ferrag, M. A., Hadjadj-Aoul, Y., Maglaras, L., Janicke, H., & Deltimple, M. (2023). SecurityBERT: A novel pre-trained language model for cybersecurity NLP. 2023 IEEE International Conference on Cyber Security and Resilience (CSR), pp. 1–6.

[13] Chismon, D., & Ruks, M. (2015). Threat intelligence: Collecting, analysing and sharing for rapid incident response. SANS Institute Reading Room.

[14] Federal Bureau of Investigation (FBI). (2023). Internet Crime Report 2022. IC3, FBI. Available: https://www.ic3.gov/Media/PDF/AnnualReport/2022_IC3Report.pdf

[15] Motlagh, F. H., Hajizadeh, M., Majd, M., Najafi, P., Cheng, F., & Meinel, C. (2024). Large Language Models in Cybersecurity: State-of-the-Art. arXiv preprint arXiv:2402.00891.

[16] Yao, S., Zhao, J., Yu, D., Du, N., Shafran, I., Narasimhan, K., & Cao, Y. (2023). ReAct: Synergizing reasoning and acting in language models. ICLR 2023. arXiv: 2210.03629

[17] Sahingoz, O. K., Buber, E., Demir, O., & Diri, B. (2019). Machine learning based phishing detection from URLs. Expert Systems with Applications, 117, 345–357.

[18] Shackleford, D. (2016). Who's Using Cyberthreat Intelligence and How? SANS Institute Survey Report.

[19] Wagner, T. D., Mahbub, K., Palomar, E., & Abdallah, A. E. (2019). Cyber threat intelligence sharing: Survey and research directions. Computers & Security, 87, 101589.

[20] IBM Security. (2023). Cost of a Data Breach Report 2023. IBM Corporation. Available: https://www.ibm.com/reports/data-breach