

AEGIS.OS: An AI-Powered, Explainable Cybersecurity Assistant for Students and Small Organizations

¹Dr. Mary Bearly E, ²Brahmadath S, ³Akshesh S M, ⁴K Karthick, ⁵Ashwin S

Department of Artificial Intelligence and Data Science, Loyola Institute of Technology and Science, Thovalai, Kanyakumari Dist, Tamil Nadu, India – 629302

Abstract

Cybersecurity threats such as malware, phishing, weak credentials, and exposed network services remain among the leading causes of security breaches, yet effective protection tools are frequently expensive, technically complex, or designed primarily for enterprise environments rather than for students, hobbyist developers, and small organizations with limited budgets and no dedicated security staff. This paper presents the design and architecture of AEGIS.OS, a lightweight, AI-powered cybersecurity assistant that unifies eight modules — Secure Authentication and Dashboard, an AI File Scanner, a URL and Phishing Detector, a Password Security Analyzer, an authorized-network Host and Port Scanner, a Website Vulnerability Checker, an AI Security Assistant, and Scan History and Report Generation — behind a single, easy-to-use web interface. What distinguishes AEGIS.OS from prior classification-only or enterprise-oriented tools is its explanation layer: SHAP-based feature attribution paired with a large language model (LLM) translates raw classifier and scan output into plain-language guidance, so a non-expert user learns not only that a file, link, password, or exposed service is risky, but why, and what to do about it. The platform is built on a React frontend, a Python (FastAPI) backend, Random Forest classifiers for file and URL risk scoring, Nmap-based host discovery for the network module, rule-based HTTP/TLS analysis for the vulnerability module, and a Supabase/PostgreSQL persistence layer. Network scanning and vulnerability assessment are scoped strictly to networks and systems the user is authorized to test, consistent with defensive, non-intrusive security analysis. This paper situates AEGIS.OS against deep-learning and explainable-AI literature for cybersecurity published between 2019 and 2022 and identifies a consistent gap: existing surveys and prototypes address classification accuracy or explainability in isolation, rarely both, and almost never within a system scoped and priced for individual students or small teams. AEGIS.OS is offered as a demonstration that meaningful, explainable, multi-domain cybersecurity protection does not require enterprise budgets or infrastructure.

Keywords — Cybersecurity; Explainable AI; SHAP; Random Forest; Phishing Detection; Malware

Detection; Password Entropy; Network Scanning; Vulnerability Assessment; Large Language Model; FastAPI

Introduction
Cybersecurity incidents are rarely the product of a single catastrophic failure; more often they accumulate from small, preventable oversights — a downloaded attachment that was never checked, a link clicked without inspection, a password reused across accounts, or a service left exposed on a network without anyone auditing it. Individually, each of these failure modes is well understood and, in principle, detectable with modest computational effort. In practice, the tools capable of that detection are unevenly distributed: enterprise security

suites offer file scanning, phishing detection, credential auditing, and network and vulnerability assessment as integrated capabilities, but they are licensed, configured, and priced for organizations with dedicated IT security teams, not for a student building a capstone project or a small business with a five-person staff.

A second, related gap concerns comprehension rather than detection. Even where free or low-cost tools exist, their output is typically a binary verdict — malicious or benign, open or closed, safe or unsafe — with little indication of why the system reached that conclusion. Two developments make it newly practical to close b

oth gaps at once. First, ensemble classifiers such as Random Forest achieve strong accuracy on structured security-relevant features (file metadata, URL lexical and host features, password composition) without requiring the scale of data or compute that deep architectures demand, while long-established tools such as Nmap provide reliable, scriptable host and port discovery. Second, explainable-AI techniques—most notably SHAP (SHapley Additive exPlanations)—can attribute a classifier's decision to specific input features, and large language models can convert that attribution, together with scan output, into a plain-language explanation a non-specialist can follow.

AEGIS.OS combines these developments into a single, lightweight platform organized around eight modules: authentication and a centralized dashboard; an AI File Scanner; a URL and Phishing Detector; a Password Security Analyzer; a Network Scanner restricted to authorized hosts; a Website Vulnerability Checker; an AI Security Assistant that renders every module's findings in plain language; and Scan History and Report Generation. All modules share a common explanation layer, so that detection and comprehension are delivered together rather than as separate concerns.

The intended audience for AEGIS.OS is deliberately narrow: individual students, early-career developers, and small organizations without a dedicated security function, rather than security operations teams already served by commercial platforms. This framing is why the system favors an explanation a non-specialist can act on over an analyst-oriented dashboard, and why the Network Scanner and Vulnerability Checker are scoped strictly to authorized targets rather than open-ended reconnaissance tools. A secondary motivation, beyond direct

protection, is security education: pairing every finding with an explanation of the underlying signal is intended to build the user's own intuition for risk, across all eight modules, rather than training the user to defer to an opaque verdict.

1. Objectives

The project is organized around one central objective — demonstrating that meaningful, explainable, multi-domain cybersecurity protection can be delivered without enterprise cost or complexity—decomposed into the following specific objectives:

- To provide secure user authentication and a centralized dashboard for monitoring security activity across all modules.
- To develop a File Scanner that identifies suspicious or malicious files using SHA-256 hash verification combined with a Random Forest classifier trained on file-level features.
- To build a URL and Phishing Detector that classifies submitted links using lexical, host-based, and structural features before a user visits the site.
- To design a Password Security Analyzer that scores password strength using entropy calculation and common-password detection, and recommends stronger alternatives.
- To provide a Network Scanner that performs host discovery and open-port identification strictly on networks the user is authorized to test, for defensive analysis only.
- To provide a Vulnerability Checker that evaluates authorized websites for missing HTTP security headers and weak SSL/TLS configuration.
- To integrate an AI Security Assistant, combining SHAP-based feature attribution with an LLM, that converts

every module's technical output into plain-language guidance and remediation steps.

- To generate user-facing scan history and security reports rather than raw, module-by-module logs.

2. Literature Survey

System	Method/Technology	Advantages	Limitations	Research Gap
Mahdavi Far & Ghorbani [1] (2019)	Survey of deep-learning architectures (autoencoders, RNNs, CNNs) applied to malware, phishing, spam and intrusion detection	Broad coverage of DL techniques across multiple threat categories; identifies open research directions	Survey only, no implementation; no unified platform; explainability barely discussed	No integration of classification output into a usable, end-user-facing security tool
Charmet et al. [2] (2022)	Literature survey of XAI methods (SHAP, LIME, rule-based) applied to malware, phishing, intrusion and fraud detection	Improves transparency and trust in AI-based security decisions; maps XAI methods to security sub-domains	Literature review only; no real-time or deployed implementation	No XAI-based tool combining multiple security domains for non-expert users
Zhan et al. [3] (2022)	State-of-the-art survey of XAI applications in cyber security (intrusion detection, malware, spam filtering)	First dedicated XAI-for-cybersecurity survey; proposes a roadmap for XAI adoption	No complete software implementation; conceptual and roadmap-oriented rather than applied	No lightweight, student-accessible implementation of the reviewed XAI techniques
Do et al. [4] (2022)	Systematic review (81 papers) taxonomizing deep-learning techniques for phishing detection	Comprehensive taxonomy of DL-based phishing detection; benchmark discussion	Limited to phishing; excludes malware/file analysis, credentials, and network-level checks	No system unifying phishing detection with file, credential, and network/vulnerability security
NIST SP 800-63B [5] (2020)	Defines password policy, entropy-	Industry-standard password guidelines	Not AI-based; provides no automated	No automated, explainable

informed guidance, MFA and authentication protocols	; widely adopted baseline for credential security	detection or explanation capability	enforcement of the guideline for end users
---	---	-------------------------------------	--

2.1 Deep Learning for Threat Classification

Mahdavi Far and Ghorbani [1] survey deep-learning architectures across malware, phishing, spam, and intrusion detection, reporting that these techniques generally outperform classical machine learning on large, high-dimensional datasets, at the cost of interpretability and the training-data volume. Few small deployments can supply. This trade-off directly informs AEGIS.OS's choice of Random Forest [6] over deep architectures for its file and URL classifiers.

2.2 Explainable AI for Cybersecurity

Charmet et al. [2] and Zhang et al. [3] independently survey XAI techniques — SHAP, LIME, rule-based explanation — applied to security classifiers, and both observe that very few surveyed systems embed an explanation layer inside a deployed, end-user-facing tool, or extend explanation beyond classifier output to scan-based findings such as open ports or missing security headers.

2.3 Phishing-Specific Detection

Do et al. [4], reviewing 81 papers on deep-learning-based phishing detection, establish a mature taxonomy of URL and content features, but confine their scope to phishing alone. NIST SP 800-63B [5] remains the standard for password policy but is a guideline document, not a system.

2.4 Synthesis

Read together, the five works in Table 1 sketch two parallel research threads that rarely intersect. The detection-accuracy thread [1], [4] optimizes classifiers against benchmark datasets, with interpretability treated as secondary. The explainability thread [2], [3] surveys XAI techniques in the abstract without embedding them

emin aworking,user-
facingsystemthatalsocoversnetwork-
orinfastructure-levelfindings.NISTSP800-
63B[5]is
a normative standard rather than prior art for an
implemented tool. AEGIS.OS's contribution is
not a novel classifier, explanation technique, or
scanning method — Random Forest [6], SHAP
[7], and Nmap-style discovery [10] are all
established — but the integration of file, URL,
credential, network, and vulnerability checks
into a single explained pipeline scoped for a
deployment context none of the surveyed works
directly target.

2.5 ResearchGap

Across the reviewed literature, three gaps recur.
First, classification accuracy and explainability
are rarely addressed together. Second, no
reviewed system unifies file-, URL-, credential-,
and network-level security checks behind a
single interface. Third, none of the reviewed
systems is scoped, evaluated, or costed for
deploymentbyasmallteamwithoutenterpriseinfrast
ructure.AEGIS.OSispositionedtocloseallthreegap
sby
combininglightweightclassifiersandstandardscan
ningtoolswithasharedSHAP-plus-
LLM explanationlayer, across a scope
deliberately sized for student and small-
organization use.

3. ExistingSystem

Current approaches fall into several categories,
each with a specific shortfall AEGIS.OS is
designed to avoid, summarized in Table 2.

ExistingA pproach	File	URL/Phis hing	Password	Network/ Vuln.	Explains Findings
Antiviruss oftware	Yes	Partial	No	No	Minimal
Password managers	No	No	Yes	No	Minimal
Phish ingbr owse r exten sions	No	Yes	No	No	No

Standalon e network/v ulnerabilit y scanners	No	No	No	Yes	Analyst- oriented
Enterp risesec urity platfor ms	Yes	Yes	Yes	Yes	Analyst- oriented, not end- user
AEGIS.O S (proposed)	Yes	Yes	Yes	Yes (aut hori zed sco pe)	Yes— SHAP+ LL M,e nd- use r- faci ng

Theconsequence,commonacrosscategories,isfrag
mentation:auserseekingfile,link,credential,andnet
work-facing protection must adopt and reconcile
output from several unrelated tools, most of
which explain their findings only to a trained
analyst, not to the person who owns the asset
being checked.

4. ProposedSystem

AEGIS.OSisanintelligent,AI-
poweredcybersecurityplatformcombiningclassica
lmachinelearning(Random Forest), standard
security tooling (SHA-256, Nmap-based
discovery, HTTP/TLS inspection), explainable
AI
(SHAP),andalargelanguagemodeltoprovidedetect
ionalongsideplain-
language,actionableguidanceacross eight
integrated modules.

4.1 UserAuthenticationandDashboard

Securesign-
inandacentralizeddashboardgivetheuserasinglepl
acetolaunchscansandreviewtheaggregate security
posture across all other modules.

4.2 AIFile Scanner

Uploaded files are hashed with SHA-256 and
matched against known-hash references;
structural metadata is passed to a Random Forest
classifier, and SHAP identifies which features
most influenced the classification.

4.3 URLandPhishing Detector

Submitted URLs are decomposed into lexical and structural features—length, IP-address hosts, HTTPS usage, suspicious keywords, domain characteristics, and direct patterns—and classified by a Random Forest model before the user visits the site.

4.4 Password Security Analyzer

Submitted passwords are analyzed for length, presence on common-password lists, and Shannon entropy, without persisting the plaintext value beyond the single analysis request.

4.5 Network Scanner

The Network Scanner performs host discovery and open-port identification using Nmap-based probing, restricted at all times to hosts and networks the user has confirmed authorization to test. This restriction is enforced in the workflow, not merely stated in documentation, consistent with responsible, defensive-only use.

4.6 Vulnerability Checker

For an authorized website, the Vulnerability Checker inspects HTTP response headers against OWASP-recommended security headers [9] and evaluates SSL/TLS configuration for common misconfigurations, returning a rule-based risk assessment.

4.7 AI Security Assistant

Every module's output passes through a shared explanation layer: SHAP quantifies each classifier feature's contribution, and rule-based findings from the Network Scanner and Vulnerability Checker are passed alongside classifier attributions to an LLM, which renders a short, non-technical explanation and a recommended next action for any of the eight modules.

4.8 Scan History and Report Generation

Every scan across all modules is logged to a unified history, from which the user can generate a consolidated, human-readable security report

rather than reviewing each module's output in isolation.

Final judgment and remediation authority remain with the user in every module; the AI layer is scoped to detection, feature attribution, and explanation, not autonomous remediation, and the Network Scanner and Vulnerability Checker are scoped to explicitly authorized targets only.

5. System Architecture

The system follows a five-layer architecture. The presentation layer, built in React, exposes forms and dashboards for all eight modules and contains no business logic. The API layer, implemented in FastAPI, routes requests to the appropriate module. The detection layer holds the Random Forest classifiers (File Scanner, URL Detector), the rule-based Password Analyzer, the Nmap-based Network Scanner, and the rule-based Vulnerability Checker. The explanation layer computes SHAP values for classifier decisions and forwards them, together with rule-based scan findings, to the LLM for plain-language rendering. A Supabase-managed PostgreSQL database serves as the persistence layer, handling authentication and storing scan history and generated reports across all modules.

5.1 Data Flow

- Step 1— Authentication: the user signs in through Supabase-backed authentication before a scan is initiated.
- Step 2— Submission: the user submits a file, URL, password, authorized host/IP, or authorized website through the React interface.
- Step 3— Routing and authorization check: FastAPI routes the request to the corresponding module; for the Network Scanner and Vulnerability Checker, an authorization confirmation is required before the scan proceeds.
- Step 4— Feature extraction/probing: the module computes the

relevant feature vector or, for network and vulnerability checks, performs the scan itself.

• Step 5—

Classification/assessment: the trained model or rule-based engine returns a risk score, open-port list, or vulnerability finding.

• Step 6—

Explanation: SHAP and/or rule attribution is passed to the LLM, which returns a plain-language explanation and recommended action.

• Step 7 — Response and persistence: the API returns the verdict and explanation to the interface and writes a record to the Supabase/PostgreSQL scan-history table.

5.2 Persistence and Security Considerations

The persistence layer stores scan records, generated reports, and user accounts, but never raw file contents, full

plaintext passwords, or unauthenticated network scan targets. The platform's own API surface follows OWASP

API Security Top 10 guidance [8]: request size limits, rate limiting on the password and network scan endpoints, and no server-side logging of plaintext credentials. These measures address AEGIS.OS's own attack surface, distinct from the threats it detects in user-submitted content or authorized targets.

6. Modules

This section details the features implemented within each of the eight modules introduced in Section 5.

6.1 User Authentication and Dashboard

Session-based authentication via Supabase, with a dashboard summarizing recent activity across all seven remaining modules.

6.2 AI File Scanner

- Secure file upload with size limits
- SHA-256 hash generation and known-hash matching
- File metadata analysis (declared vs. actual type, embedded executables)
- Random Forest malware classification with ar

ported risk level

- SHAP-based explanation of the classification

6.3 URL and Phishing Detector

- URL format and length analysis
- Detection of IP addresses used in place of hostnames
- Suspicious-keyword and brand-impersonation checks
- HTTPS usage and domain-characteristic checks
- Redirect-chain pattern analysis

6.4 Password Security Analyzer

- Password length evaluation
- Common-password and pattern-based blocklist matching
- Shannon entropy estimation

6.5 Network Scanner (Authorized Targets Only)

- Host discovery
- IP address detection
- Open-port identification
- Service/version detection on discovered ports
- Basic network information summary

Scanning is performed only on networks the user has explicitly confirmed authorization to test, consistent with defensive, non-intrusive cybersecurity analysis and applicable computer-misuse law.

6.6 Vulnerability Checker (Authorized Targets Only)

- HTTP security header checking (e.g., Content-Security-Policy, X-Frame-Options) against OWASP guidance [9]
- SSL/TLS configuration checking
- Basic security-misconfiguration detection
- Rule-based vulnerability risk assessment

6.7 AI Security Assistant

- Plain-language explanation of findings from all seven other modules
- Recommended remediation steps
- General security-awareness guidance

- Beginner-friendly framing of technical findings

6.8 Scan History and Report Generation

- Unified, cross-module scan history
- Consolidated, human-readable security report generation

7. Algorithms

7.1 Random Forest Classification

Random Forest [6] trains multiple decision trees on bootstrapped samples and aggregates predictions by majority vote. It was selected over deep-learning alternatives for the File Scanner and URL Detector because it performs well on the structured, low-dimensional feature sets used here without requiring large training corpora, and produces per-feature importance scores that integrate naturally with SHAP-based explanation.

7.2 SHA-256 Hashing

SHA-256 produces a fixed-length digest of a file's contents, used for integrity verification and to check a file against known-malicious hash references without persisting the file itself.

7.3 Password Entropy

Password strength is estimated as $E = L \times \log_2(N)$, where L is password length and N is the character-space size in use. Higher entropy indicates greater resistance to brute-force guessing.

7.4 Host Discovery and Port Scanning

The Network Scanner uses Nmap-style probing [10] to enumerate live hosts and open ports on an authorized target range, reporting service banners where available; this is a well-established discovery technique rather than a novel contribution, applied here defensively and with the same authorization restriction described in Section 7.5.

7.5 Rule-Based Vulnerability Analysis

The Vulnerability Checker evaluates an

authorized website's HTTP response against a checklist of expected security headers [9] and its TLS handshake against known-weak configurations, producing a deterministic, rule-based risk rating rather than a learned classification.

7.6 Explainable AI via SHAP

SHAP [7] attributes a classifier's prediction to its input features using a game-theoretic approach adapted from Shapley values. In AEGIS.OS, SHAP output — and, for the Network Scanner and Vulnerability Checker, the underlying rule that fired — is the direct input to the LLM explanation step, reducing the risk of the LLM fabricating an explanation unsupported by the underlying finding.

8. Tools and Technologies

- Frontend: React.js, Vite, JavaScript, Tailwind CSS, React Router, Axios
- Backend: Python, FastAPI, Uvicorn, Pydantic
- Database and Auth: Supabase, PostgreSQL
- AI/ML: scikit-learn (Random Forest), pandas, NumPy, joblib, SHAP
- Security: SHA-256, Nmap, SSL/TLS analysis, HTTP header analysis
- Development and testing: VS Code, Git/GitHub, Swagger UI, Postman, Pytest
- Reports and AI: ReportLab (PDF report generation), LLM-based security explanation

9. Results and Discussion

At the current stage of development, AEGIS.OS has been validated functionally rather than through a formal accuracy benchmark: each of the eight modules accepts input through the React interface, performs its detection, scan, or check, returns a plain-language explanation, and persists the result to the unified scan history. This confirms the intended pipeline functions end to end across all modules,

including the authorization gate on the Network Scanner and Vulnerability Checker.

A quantitative evaluation — classifier accuracy against labeled malware and phishing datasets, and measured false-positive/false-negative rates for the Network Scanner and Vulnerability Checker against known-configuration test hosts — has not yet been conducted and is reported here as future work rather than a completed result. Presenting specific accuracy figures without that evaluation would overstate the current state of the system; the honest claim at this stage is architectural and functional completeness across all eight modules, not measured detection performance.

9.1 Functional Verification

Table 3 summarizes the functional checks performed per module group during development — correctness checks on the pipeline rather than statistical accuracy measurements against a labeled evaluation set.

Module Group	Test Performed	Observed Behavior
File Scanner	Upload of a known-benign document and a known-hash test file	Benign file passed with low risk score; known-hash file flagged with matching reference cited in the explanation
URL/ Password	Submission of a legitimate domain and a synthetic phishing-pattern URL; a common password and a high-entropy password	Legitimate URL and strong passwords scored low risk; synthetic phishing URL and common password correctly flagged with
		SHAP/entropy detailed
Network/Vulnerability	Scan of a locally hosted, self-authorized test server with a deliberately open non-standard port and a missing security header	Open port and missing header both correctly identified and explained; scan of a non-authorized address correctly blocked by the authorization gate

These checks confirm the pipeline, including the authorization gate, behaves as designed on representative inputs; they do not substitute for the labeled-

dataset and test-infrastructure evaluation identified as future work.

10. Ethical and Responsible Use

Including a Network Scanner and Vulnerability Checker in a student-built tool carries obligations beyond the othersix modules. AEGIS.OS enforces authorization at the workflow level — as a target must be confirmed as owned or explicitly authorized by the user before the Network Scanner or Vulnerability Checker will run — rather than relying on a terms-of-use statement alone. This mirrors the restriction stated in the project's own design documentation and is treated as a hard function requirement, not a configurable option. Consistent with this scope, the modules are positioned for defensive, self-assessment use (a student auditing their own home lab or a small organization auditing its own authorized infrastructure), not for scanning third-party systems without consent, which may violate computer-misuse law regardless of intent. Section 12 below identifies formal security testing of this authorization gate as unfinished work, and readers evaluating this paper for a security-relevant contribution should weight that gap accordingly.

11. Limitations

Three limitations are stated explicitly. First, no formal accuracy evaluation against labeled datasets or known-configuration test infrastructure has yet been performed for any module. Second, the LLM explanation step is constrained to rendering SHAP or rule-based attribution rather than reasoning independently, which bounds explanation quality to the quality of the underlying attribution, itself not yet independently validated. Third, the Network Scanner and Vulnerability Checker carry real legal and ethical exposure if the authorization gate

described in Sections 5.5, 5.6, and 7.5 is bypassed or misconfigured; this gate is a functional requirement of the system, not an optional safeguard, and its correct enforcement has not yet been subjected to formal security testing of its own.

12. Conclusion and Future Scope

AEGIS.OS demonstrates that a small student team can deliver a cybersecurity tool that is simultaneously lightweight, explainable, and unified across file, link, credential, and authorized network/vulnerability risk, without the cost or complexity of enterprise security platforms. By pairing Random Forest classification and standard scanning tools with a SHAP-plus-LLM explanation layer, the platform addresses a gap identified consistently across the surveyed literature: systems that detect accurately rarely explain their findings in terms a non-expert can act on.

Future work includes the formal accuracy and usability evaluation described in Section 10, independent security testing of the Network Scanner's authorization gate described in Section 12, and expansion of notification channels beyond in-application delivery, without requiring a redesign of the detection-plus-explanation architecture presented here.

12.1 Phased Roadmap

The following phased plan is proposed for work beyond the scope of this paper, ordered by dependency:

- Phase 1 (current): Eight-module pipeline (Sections 5–9) with SHAP/rule-based attribution and LLM explanation across all modules.
- Phase 2: Formal evaluation — precision, recall, and F1-score for the File Scanner and URL Detector against labeled datasets; false-positive/false-negative measurement for the Network Scanner and Vulnerability Checker against known-configuration test hosts; and independent penetration testing of the authorization gate described in Section 11.
- Phase 3: A small-sample user comprehension study comparing explained versus

unexplained output across modules, addressing the evaluation gap identified in Section 10.

- Phase 4: Notification-channel expansion (email, SMS) beyond current in-application delivery, and a lightweight administrative view for small-organization deployments managing scans across multiple users and multiple authorized targets.

AUTHOR CONTRIBUTIONS

Dr. Mary Bearly supervised the project and reviewed the manuscript. Brahmada S, Akshay S, K Kartick, and Ashwin S jointly contributed to system design, implementation, and evaluation across the eight modules. All authors reviewed and approved the final manuscript.

ACKNOWLEDGMENT

The authors thank the Department of Artificial Intelligence and Data Science, Loyola Institute of Technology and Science, for the guidance and resources that supported this work.

CONFLICT OF INTEREST

The authors declare no conflict of interest. AEGIS.OS is an academic capstone project and is not, at the time of writing, a commercial product.

REFERENCES

- [1] S. Mahdavi Far and A. A. Ghorbani, "Application of deep learning to cybersecurity: A survey," *Neurocomputing*, vol. 347, pp. 149–176, 2019.
- [2] B. Charret, H. Chieze, R. Alonso, F. Migot, S. Heitor, N. Boulahia-Cuppens, and F. Cuppens, "Explainable artificial intelligence for cybersecurity: a literature survey," *Annals of Telecommunications*, vol. 77, pp. 789–812, 2022.
- [3] Z. Zhang, H. Al Hamadi, E. Damiani, C. Y. Yeun, and F. Taher, "Explainable Artificial Intelligence Applications in Cyber Security: State-of-the-Art in Research," *IEEE Access*, vol. 10, pp. 93104–93139, 2022.
- [4] N. Q. Do, A. Selamat, O. Krejcar, E. Herrera-Viedma, and H. Fujita, "Deep Learning for Phishing Detection: Taxonomy, Current Challenges and Future Directions," *IEEE Access*, vol. 10, pp. 36429–36463, 2022.
- [5] P. A. Grassi, M. Garcia, and J. Fenton, "Digital

Identity Guidelines: Authentication and Lifecycle Management," NIST Special Publication 800-63B, National Institute of Standards and Technology, 2020.

[6] L. Breiman, "Random Forests," Machine Learning, vol. 45, no. 1, pp. 5–32, 2001.

[7] S. M. Lundberg and S.-I. Lee, "A Unified Approach to Interpreting Model Predictions," in Advances in Neural Information Processing Systems (NeurIPS), vol. 30, 2017.

[8] OWASP Foundation, "OWASP API Security Top 10," 2023. [Online]. Available: <https://owasp.org/API-Security/>

[9] OWASP Foundation, "OWASP Top 10 Web Application Security Risks," 2021. [Online]. Available: <https://owasp.org/www-project-top-ten/>

[10] Nmap Project, "Nmap Network Scanning: The Official Nmap Project Guide to Network Discovery and Security Scanning." [Online]. Available: <https://nmap.org/book/>

[11] National Institute of Standards and Technology (NIST), "Cybersecurity Framework (CSF) 2.0," 2024.