JKSCI

한국컴퓨터정보학회논문지
Journal of The Korea Society of Computer and Information
Vol. 31 No. 1, pp. 1-12, January 2026
https://doi.org/10.9708/jksci.2026.31.01.001

A Proposal of an LSTM-Based Machine Learning and Hybrid Application Security Testing Architecture for Web Vulnerability Detection

Ha Young Kim*, Seong-Cho Hong**, Ah Reum Kang***

*M.S. Student, Dept. of Cyber Security, Pai Chai University, Daejeon, Korea
**Research Professor, Smart ICT Convergence HRD Center, Pai Chai University, Daejeon, Korea
***Professor, Dept. of Cyber Security, Pai Chai University, Daejeon, Korea

[Abstract]

This study proposes an HAST architecture that integrates a machine learning–based LSTM model with SAST and DAST to address the growing number of vulnerabilities in web application environments. An analysis of previous studies reveals several limitations in existing web vulnerability detection approaches, including the lack of standardized datasets, limited domain generalization, and insufficient responsiveness to real-time attack scenarios. To overcome these challenges, the proposed architecture combines LSTM-based request sequence analysis with a unified SAST–DAST pipeline. The proposed HAST structure supports real-time request detection, coordinated static and dynamic analysis, and a retrainable expansion mechanism, enabling a stepwise response to evolving web application environments and emerging attack patterns. The results are expected to support the development of an integrated response framework for web vulnerability detection and to provide a structural design foundation for future research.

▸ **Key words**: Web Vulnerability, LSTM, SAST, DAST, HAST

[요 약]

웹 애플리케이션 환경에서 지속적으로 증가하는 취약점을 탐지하기 위해, 본 연구는 머신러닝 기반 LSTM 모델과 SAST 및 DAST를 결합한 HAST 보안 아키텍처를 제안하였다. 기존 선행 연구 분석을 통해, 웹 취약점 탐지에 대한 표준화된 데이터 셋 부재, 도메인 일반화 한계, 실시간 탐지 대응력 부족 등의 한계를 지니고 있음을 확인하였다. 이러한 한계를 고려하여 본 논문에서는 머신러닝 기반 LSTM 기반 요청 시퀀스 분석과 SAST와 DAST를 하나의 파이프라인으로 결합하는 HAST 통합 구조를 설계하였다. 제안한 아키텍처는 실시간 요청 탐지, 정적·동적 분석 연계, 재학습 가능한 확장 구조를 포함함으로써 웹 애플리케이션 환경 변화와 새로운 공격 패턴에 점진적으로 대응할 수 있는 구조적 방향성을 제시하였다. 연구의 결과는 웹 취약점 탐지를 위한 통합적 대응 구조와 향후 연구를 위한 설계 기반을 제시하는 데 도움이 될 것이다.

▸ **주제어**: 웹 취약점, LSTM, SAST, DAST, HAST

• First Author: Ha Young Kim, Corresponding Author: Ah Reum Kang
*Ha Young Kim (rlagkdud3565@naver.com), Dept. of Cyber Security, Pai Chai University
**Seong-Cho Hong (scv.hong@pcu.ac.kr), Smart ICT Convergence HRD Center, Pai Chai University
***Ah Reum Kang (armk@pcu.ac.kr), Dept. of Cyber Security, Pai Chai University
• Received: 2025. 10. 22, Revised: 2025. 12. 22, Accepted: 2025. 12. 29.

http://www.ksci.re.kr pISSN:1598-849X | eISSN:2383-9945

I. Introduction

현대 사회에서 인터넷과 웹 애플리케이션 사용이 급속도로 증가하면서 웹 보안 위협도 함께 심각해지고 있다[1]. 웹 보안을 위협하는 웹 애플리케이션의 취약점 중 지속적으로 SQLi(SQL Injection), XSS(Cross-Site Scripting), CSRF(Cross-Site Request Forgery) 등의 공격이 주요 위협으로 지목되고 있다. 이러한 웹 공격은 기업과 사용자에게 정보 유출, 서비스 장애, 데이터 무결성 훼손 등의 심각한 피해를 초래할 수 있어 이를 효과적으로 탐지하고 대응할 수 있는 방안 마련이 시급한 과제로 떠오르고 있다[1].

이러한 상황에서 최근 웹 보안 위협에 대한 대응책으로 머신러닝 기반의 웹 취약점 탐지 기법들이 각광받고 있다[2]. 머신러닝 기반의 웹 취약점 탐지 기법은 방대한 웹 트래픽 속에서 정상 행위와 비정상 행위를 자동으로 구분하고, 기존 시그니처(Signature) 기반 보안 시스템이 탐지하지 못하는 Zero-day 공격에 대해서도 높은 탐지 가능성을 제시함으로써 차세대 웹 보안 기술로 주목받고 있다[3][4][5]. 또한 웹 애플리케이션 보안 테스트 방법론을 통해 데이터 유출 및 침해와 같은 사이버 공격을 방지하고, 취약점 식별 및 소프트웨어의 전반적인 보안을 강화하고 있다. 그중 가장 널리 사용되는 방법론은 SAST(Static Application Security Testing), DAST(Dynamic Application Security Testing)이다. 이들을 조합하거나 단계적으로 함께 적용하는 것이 잠재적인 사이버 공격에 대한 보호와 보안을 강화할 수 있다[6].

따라서 본 연구에서는 효과적인 웹 애플리케이션 취약점 탐지를 위해 LSTM(Long Short-Term Memory) 모델과 SAST, DAST을 결합한 HAST(Hybrid Application Security Testing) 방법론을 제안하고자 한다. 제안 모델은 이론적으로 실시간 탐지 능력과 실제 웹 서비스 환경에서의 높은 일반화 성능을 동시에 확보할 수 있는 구조이다. 구체적으로 기존의 선행 연구들을 개선하여 머신러닝과 보안 테스트 방법론을 활용한 아키텍처를 제안하며 보다 효과적인 웹 취약점 탐지 방안에 대해 제시하고자 한다.

이에 본 논문의 구성은 다음과 같다. 2장에서는 웹 취약점 탐지와 머신러닝 기반 보안 연구에 관한 선행 연구를 검토하고, 3장에서는 LSTM 기반의 탐지 모델과 SAST·DAST를 통합한 HAST 아키텍처의 설계 요소를 제시한다. 4장에서는 제안 모델의 처리 흐름과 각 단계별 구성 요소를 상세히 기술하며, 5장에서는 연구의 결론과 함께 제안 아키텍처의 의의와 한계를 논의한다.

II. Related Works

2.1. Web Application Vulnerability

웹 애플리케이션(Web Application)이란 인터넷 또는 인트라넷과 같은 네트워크를 통해 액세스 되는 응용 프로그램을 의미한다. 주로 웹 브라우저 내 또는 웹 브라우저가 제어 가능한 환경에서 실행되며, 자바스크립트와 같은 웹 브라우저가 실행 가능한 프로그래밍 언어를 사용하여 HTML과 같은 마크업 언어와 결합하여 만들어진다[7][8]. 이러한 웹 애플리케이션은 웹 브라우저를 통해 접근하기 때문에 다양한 플랫폼에서 호환이 가능하며, 별도의 배포나 설치 과정이 필요하지 않아 사용이 편리하고, 업데이트나 유지보수가 용이하다는 장점을 가지고 있다. 다만 웹 애플리케이션은 인터넷 환경에서 주로 사용되기 때문에 웹 애플리케이션의 보안 취약점을 악용한 공격의 대상이 될 수 있다[1].

보안 약점은 보안 취약점이 될 수 있는 일반적인 형태를 말하는데[9][10], 대표적인 주요 보안 약점 목록으로 OWASP Top 10과 CWE/SANS Top 25를 꼽을 수가 있다[10]. 먼저 OWASP(Open Worldwide Application Security Project)가 주관하는 보안 프로젝트인 OWASP Top 10은 웹 보안 동향을 이해하고 대응 전략을 수립하기 위해 전 세계에서 많이 참고 되는 목록이다[1]. 전 세계적으로 발생 빈도, 상대적 공격 가능성, 탐지 가능성 및 영향도에 따라 보안에 가장 중대한 영향을 미치는 10가지 취약점이 선정되며[1][11], 가장 최신 목록의 구체적인 내용은 아래 Table 1과 같다[12].

Table 1. OWASP Top 10:2021

OWASP Top 10:2021
A01 : Broken Access Control
A02 : Cryptographic Failures
A03 : Injection
A04 : Insecure Design
A05 : Security Misconfiguration
A06 : Vulnerable and Outdated Components
A07 : Identification and Authentication Failures
A08 : Software and Data Integrity Failures
A09 : Security Logging and Monitoring Failures
A10 : Server-Side Request Forgery

CWE(Common Weakness Enumeration)는 미국 Department of Homeland Security의 지원 하에 MITRE에서 관리하고 있으며, 실제적인 보안 약점으로 944개의 항목을 제공하고 있다[10]. 또한 CWE는 소프트

웨어 취약점 중 가장 위험한 소프트웨어 공통 보안 약점 25가지 취약점 목록인 CWE/SANS Top 25를 선정하여 발표하고 있다[13]. 2009년부터 25개의 주요 보안 약점을 선발하여 발표하고 있으며 2024년에 가장 최근 버전이 발표되었다[10]. 아래 Table 2는 발표된 25개의 항목의 순위를 나타낸 것이다[14].

Table 2. CWE/SANS Top 25:2024

순위	CWE-ID	취약 항목
01	79	Cross-site Scripting
02	787	Out-of-bounds Write
03	89	SQL Injection
04	352	CSRF
05	22	Path Traversal
06	125	Out-of-bounds Read
07	78	OS Command Injection
08	416	Use After Free
09	862	Missing Authorization
10	434	Unrestricted Upload of File with Dangerous Type
11	94	Code Injection
12	20	Improper Input Validation
13	77	Command Injection
14	287	Improper Authentication
15	269	Improper Privilege Management
16	502	Deserialization of Untrusted Data
17	200	Exposure of Sensitive Information to an Unauthorized Actor
18	863	Incorrect Authorization
19	918	SSRF
20	119	Improper Restriction of Operations within the Bounds of a Memory Buffer
21	476	NULL Pointer Dereference
22	798	Use of Hard-coded Credentials
23	190	Integer Overflow or Wraparound
24	400	Uncontrolled Resource Consumption
25	306	Missing Authentication for Critical Function

한편, 글로벌 웹 서비스 보안업체인 IMPERVA에서 사이버 공격을 시도한 출처별 통계자료(2025년 8월)를 통해 웹 서비스 사용자를 대상으로 사이버 공격 시도가 가장 많이 발생하고 있다고 발표하였다[15]. 이처럼 현대에 이르러서도 여전히 웹 애플리케이션에 대한 공격 시도가 증가하고 있으며, 그만큼 웹 애플리케이션 서비스 사용자 또한 지속적으로 증가하고 있음을 확인 할 수 있다. 다음의 Fig. 1은 사이버 공격 출처별 통계자료(2025년 8월)이다[15].

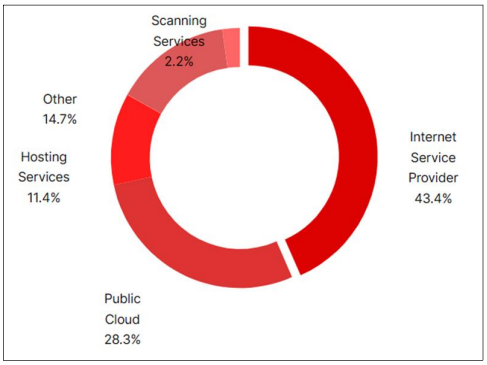

Fig. 1. Cyber Attack Attempt Rates by Source

이러한 웹 애플리케이션 공격 중 탐지 측면에서 변종 공격이나 Zero-day 공격, 자동화 공격 등은 쉽게 탐지하기가 어렵다[3]. 기술적 측면에서도 취약점 다양성, 언어/프레임워크별 특성처럼 계속해서 진화하는 취약점과 기술 구조에 따라 탐지 및 대응이 어렵다는 문제가 있다[16].

실제로 한국인터넷진흥원 통계에 따르면 악성코드 은닉 사이트 탐지는 2020년 6,034건, 2021년 7,043건, 2022년 13,661건, 2023년 12,731건, 2024년 13,967건으로 계속하여 증가하고 있으며, 더불어 침해사고 신고접수 또한 2020년부터 지속적으로 증가하고 있다[17]. 아래 Fig. 2는 최근 5년간 발생한 해킹사고 건수 그래프이다[17].

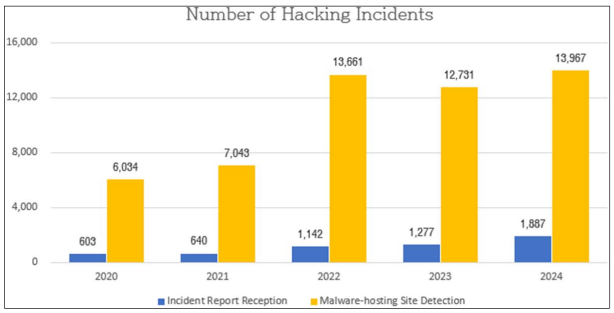

Fig. 2. Graph of the Number of Hacking Incidents

이러한 수치들을 종합해 볼 때, 공격자들이 알려진 취약점을 빠르게 무기화하고, 원격 작업용 소프트웨어와 클라우드 애플리케이션들의 취약점에 대한 위협도 증가할 것이라고 보고 있다. 특히 Zero-day 취약점뿐만 아니라 이미 알려졌으나 패치되지 않은 취약점인 n-day 취약점까지도 큰 위협이 될 가능성이 높다[18].

웹 애플리케이션 취약점 관련 OWASP Top 10:2010의 여러 버전에서는 지속적으로 SQLi, XSS, CSRF 등의 공격이 주요 위협으로 지목되어 왔으며, 소프트웨어 취약점 관련 CWE/SANS Top 25에서도 XSS, 삽입공격, 인증 우회 등의 항목이 주요 소프트웨어 결함으로 선정되어 왔다.

이러한 웹 공격을 효과적으로 탐지하고 대응할 수 있는 방안으로 최근에는 머신러닝 기반의 웹 취약점 탐지 기법이 대안으로 주목받고 있으며, 이는 방대한 웹 트래픽 속에서 기존의 시그니처 기반 보안 체계가 탐지하지 못하는 Zero-day 공격을 식별할 수 있는 가능성을 보여주고 있다.

2.2. Machine Learning-based Vulnerability Detection and Limitation

웹 공격 탐지 성능을 향상시키기 위한 다양한 머신러닝 기반 선행 연구에서는 주로 Random Forest, Isolation Forest, Support Vector Machine(SVM), Convolutional Neural Network(CNN), Recurrent Neural Network(RNN), Long Short-Term Memory(LSTM) 등의 기법이 활용되어 왔다[2][5-6][19-29].

선행 연구들의 머신러닝 기법을 크게 나누어 보자면 지도학습과 비지도학습과 같은 학습 패러다임으로 분류된다고 볼 수 있다. 먼저 지도학습(Supervised Learning)은 컴퓨터가 입력 값과 그에 따른 출력 값이 있는 데이터를 이용하여 주어진 입력에 맞는 출력을 찾는 학습 방법이다[30]. 지도학습기가 하는 작업은 훈련 데이터로부터 주어진 데이터에 대해 예측하고자 하는 값을 올바로 추측해 내는 것이다[3]. 주로 이메일의 스팸 여부 분류, 소셜 미디어 공유 점수 및 성과 점수 예측, 이미지 인식 등에 활용되고 있다[3]. 반면에 비지도학습(Unsupervised Learning)은 별도의 학습용 데이터를 구축하는 것이 아니라 데이터 자체를 분석하거나 군집하면서 학습하는 방법이다[31]. 주로 구매 행동에 따른 고객 그룹화, 사진 목록에서 비슷한 얼굴로 그룹화, 고객 데이터에서 연관성 식별 등에 활용되고 있다[3]. Fig. 3은 앞서 설명한 지도학습과 비지도학습을 도식화한 그림이다[3].

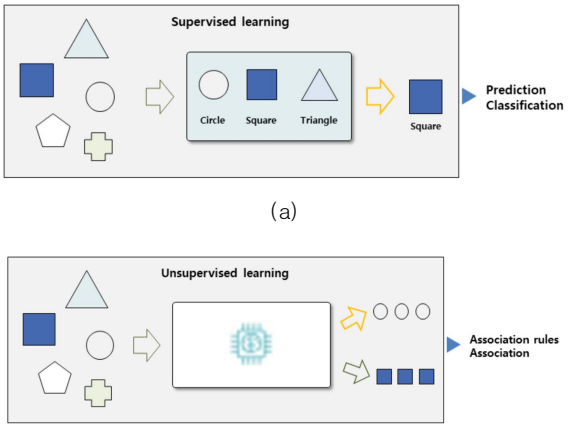

Fig. 3. (a) Supervised Learning (b) Unsupervised Learning

각 알고리즘은 웹 공격 탐지에 활용될 때 서로 다른 방식으로 특징을 추출하고 공격 패턴을 식별한다. 지도학습 계열에서는 Random Forest, SVM, CNN, RNN, LSTM 등이 주로 활용되었다. Random Forest는 다양한 입력 특성의 조합을 기반으로 공격 여부를 다면적으로 판단할 수 있으며 다수의 결정 트리를 앙상블 하여 높은 분류 정확도를 제공하지만, 특성 공학과 대량의 레이블 데이터 확보가 필요하다. SVM은 초평면 기반의 이진 분류를 통해 정상·비정상 요청을 명확히 구분하며 고차원 공간에서 우수한 분류 성능을 보이지만, 대규모 데이터 셋에 적용할 경우 계산 비용이 커진다. CNN은 URL 또는 로그 문자열 내의 지역적 패턴을 필터 기반으로 추출하여 SQL Injection이나 XSS와 같이 특정 키워드 반복이나 특수문자 패턴이 뚜렷한 공격에 효과적이나 입력 표현 방식에 따라 성능 차이가 발생한다. RNN과 LSTM은 웹 요청 흐름의 순차적 구조를 분석하여 긴 문장 기반 공격 벡터나 반복적 페이로드 구조를 탐지하는 데 유리하다. 시퀀스 기반 요청 패턴을 학습하는 데 적합하며, 특히 LSTM은 긴 시퀀스에서 장기 의존성을 효과적으로 학습할 수 있어 HTTP 페이로드나 URL 시퀀스와 같은 연속적 데이터에서 강점을 보인다. 이렇듯 지도학습 기반의 알고리즘은 높은 정확도를 보이지만 실시간 탐지, 높은 비용 등에 어려움을 가지고 있다는 한계를 가지고 있다.

비지도학습 계열에서는 Isolation Forest와 Auto Encoder가 대표적으로 활용된다. Isolation Forest는 트리 기반의 격리 구조로 비정상 요청을 빠르게 탐지하고, Auto Encoder는 입력 재구성 오류를 활용해 기존에 관찰되지 않은 새로운 공격 패턴을 식별할 수 있다. 이들은 레이블이 없는 환경에서도 이상치와 Zero-day 공격을 탐지하는 것이 유리하다는 장점이 있으나, 오탐률(False Positive Rate)이 높다는 한계가 존재한다. 학습 유형에 따른 머신러닝 알고리즘을 분류해 보면 아래의 Table 3과 같다.

앞서 설명한 알고리즘들은 특정 취약점이나 데이터 유형에서 높은 탐지 성능을 보여 왔으나, 여전히 표준화된 데이터 셋 부재, 도메인 종속성, 실시간 탐지 제약, Zero-day 공격 대응 부족이라는 공통적인 한계를 지니고 있다. 따라서 단일 모델만으로는 OWASP Top 10, CWE/SANS Top 25의 전 범위를 안정적으로 탐지하기 어렵다는 한계가 존재한다.

Table 3. Algorithm Characteristics According to Learning Type

Learning type	Algorithm	Key Features	Utilize Web Attack Detection
Supervised Learning	Random Forest	Combination of Multiple Decision Trees	Attack Classification Using Various Input Characteristics
	SVM	Hyperplane-based Binary Classification	Normal/Attack Request Classification
	CNN	Filter based Feature Extraction	Detect Attack Patterns in Logs and URLs
	RNN	Sequential Data Processing	Web Request Sequence-based Detection
	LSTM	Long-term Dependent Memory	Detecting Long Statement-based Attack Vectors
Unsupervised Learning	Isolation Forest	Outlier Detection Model	Detecting Abnormal Traffic and New attacks
	Auto Encoder	Restoration Error-based Anomaly Determination	Detect New Attacks or Anomalies through High Restoration Errors

선행 연구들을 검토한 결과, 머신러닝 기반의 탐지 모델을 만들 때 중요한 성공 요소가 세 가지 유형의 데이터 셋(실제 로그 기반, 표준 공개 데이터, 생성 데이터)임을 확인하였다[5][10][21]. 먼저 실제 로그 기반 데이터 셋은 실제 운영 환경에서 수집된 침해사고 로그·웹서버 공격 로그·IPS 이벤트 등을 뜻한다. 이 데이터 셋은 현실 반영도가 높고 머신러닝 학습에 적합하지만, 데이터 셋의 노후화, 범위 한정, 표준화 부족 등의 한계가 존재했다[4][20-21][32-34]. 반면 WebGoat, Juice Shop, bWAPP 등 표준 공개 데이터는 재현성과 실습성 측면에서 장점이 있으나 최신 공격 패턴·실시간 로그를 충분히 반영하지 못한다는 한계가 존재한다[35-38]. 최근에는 ChatGPT 등을 이용한 생성 데이터로 현실성 높은 시뮬레이션을 확보하려는 시도도 보고되었으나, 생성 데이터가 실제 운영 로그를 완전히 대체하기에는 아직까지 한계가 존재하는 것으로 나타났다[40-44]. 아래 Table 4는 각 데이터 셋 유형에 따른 장점과 한계를 비교한 표이다.

Table 4. Comparing the Advantages and Limitations of Different Data Set Types

Data Set	Previous Studies	Data Source	Advantages	Limitations
Real-World Log Bsed	[4] [20-21] [32-34]	◦ Breach Incident Log ◦ Web Server Attack Log	◦ High Reflection of Reality ◦ Suitable for Machine Learning	◦ Dataset Aging ◦ Limited Scope ◦ Lack of Standardization
Standard Open Data	[35-38]	◦ WebGoat ◦ Juice Shop ◦ bWAPP	◦ Standardization ◦ Reproducibility ◦ Practice oriented Data	◦ Real-time Log not Reflected ◦ Limited to the Latest Attack
Generation Data	[40-44]	◦ Self-Generated ◦ GPT-3.5	◦ Highly Realistic simulation ◦ Diverse Activities Comprehensive	◦ Lack of Data in the Real-World

이와 같은 비교 분석 결과를 종합해 보면, 표준화된 실제 운영 환경 기반 데이터 셋의 부재가 머신러닝 기반 탐지 모델의 도메인 일반화와 현실 적용성에 중요한 제약으로 나타났다. 본 연구는 이러한 한계를 해결하기 위해 실제 운영 로그를 고려한 데이터 수집·정제 전략과 함께, 비정형 웹 데이터의 의미를 잘 포착할 수 있는 머신러닝 기반 LSTM을 활용하고자 한다.

2.3. Web Application Security Testing Methodologies : SAST and DAST

웹 애플리케이션은 사용자와 데이터·서비스를 연결하는 수단으로서 접근성이 높기 때문에[28], 노출도가 높아 사이버 공격자의 주요 표적이 되어 약점을 악용하여 프라이버시를 침해하려는 시도의 대상이 되었다[45][46]. 따라서 소프트웨어 생명주기의 각 단계에서 소프트웨어 보안을 평가하는 것은 필수적으로 되었다. 소프트웨어 보안을 평가하는 방법과 수단은 여러 가지가 있으며, 그 중 애플리케이션 보안 테스트(Application Security Testing)의 활용이 두드러진다.

보안 테스트를 수행하는 도구(프로그램)는 개발자가 애플리케이션의 보안을 확보하기 위해 가장 널리 사용하는 자원 중 하나이다[47]. 또한 시스템을 손상시킬 수 있는 취약점으로 이어질 수 있는 보안 약점을 탐지하거나 이를 방지하는 데 도움을 줄 수 있다[48]. 소프트웨어를 개선하기 위한 접근 방식은 다양하며, 소프트웨어 보안 테스트를 수행하는 도구를 분류하는 가장 일반적인 방법은 해당 도구가 애플리케이션을 전체 단위로 취급하여 외부에서 평가

하는 블랙박스(black-box) 접근인지와 소스 코드에 접근하여 소프트웨어의 내부 동작을 검사하는 화이트박스(white-box) 접근인지의 여부이다[48].

보안테스트 방법인 SAST와 DAST에 대하여 살펴본 결과, SAST는 소스 코드 내부를 검사하는 화이트박스 테스트 방식이며, 소프트웨어 개발 생명주기에서 취약점을 조기에 식별하기 위해 보통 개발 초기 단계에서 사용된다. 또 개발을 효과적이고 신뢰할 수 있도록 자동화하여 처리해주는 CI/CD의 이점(오류 조기 발견, 배포 시간 단축 등)을 지속적 통합단계에서 수행하게 된다. 그리고 컴파일 되지 않은 소스 코드 분석을 기반으로 하여 근본적인 수준에서 취약 패턴과 코딩 결함을 포괄적으로 찾아낼 수 있다. SAST 기술의 장점은 SQLi, XSS와 같은 심각한 취약점을 식별하며, 자동화를 통해 코드 내의 결함 위치(예: 라인 번호)를 즉시 비교적 정확하게 피드백해 준다. 대표적인 도구로는 SonarQube와 FindSecBugs 등이 있다[22-23][27][35][49-52]. 단점으로는 식별된 보안 문제가 실제 취약점인지 확인하기 어렵고, 정적인 코드베이스만 분석하기 때문에 런타임 문제나 설정 취약점을 놓치기도 하며[22] 자동으로 발견할 수 있는 취약점의 비율이 제한적이다[53]. SAST 도구를 선정하여 취약점 탐지를 진행한 결과[54], 실제 환경의 취약점 중 단 12.7%만 탐지할 수 있었다. 특히 자원제어 관련 취약점과 입출력 취약점은 거의 탐지하지 못하였다는 것으로 보았을 때, 런타임 환경에서만 드러나는 결함은 검증하기 어렵다는 한계가 존재했다.

DAST는 실행 중인 애플리케이션을 외부에서 관찰되거나 악용될 수 있는 취약점을 식별하는 블랙박스 테스트 방식으로 언어 및 플랫폼 독립성이 높다. 실제 공격과 유사한 방식을 시뮬레이션하여 인증 우회, 세션 관리 결함, 런타임 입력 검증 문제, 서버 설정 오류 등과 같은 운영 환경 중심의 취약점을 검출하여 애플리케이션의 보안 상태를 더욱 포괄적으로 평가할 수 있게 한다[23]. DAST는 보통 테스트 환경에 소프트웨어가 배포된 이후에 수행되어 잠재적 취약점에 대한 데이터를 수집하여 전체 애플리케이션의 보안성을 평가하고 CI/CD 환경을 자동화하여 운영 환경을 보완하는 용도로 사용할 수 있다[22][24-27][43][55-56]. 대표적인 도구로는 OWASP ZAP, Burp Suite 등이 있다[22]. DAST 단점으로는 내부 코드 구조를 알 수 없기 때문에 원인 파악이 어려우며, 공격 시나리오 반복 테스트로 인해 SAST보다 테스트 시간이 오래 걸린다. 또 테스트하기 위한 런타임 환경이 필요하여 초기 설정에 부담이 될 수 있다. 아래 Table 5는 선행 연구를 바탕으로 SAST와 DAST의 주요 특징과 장단점을 비교 정리한 것이다.

Table 5. Characteristics and Strengths/Limitations of SAST and DAST

Division	Previous Studies	Characteristic
SAST	[22-23] [35] [49-52]	◦ Focus on Static Code Analysis ◦ Minimize Performance Degradation in CI/CD Environments ◦ Unverified Vulnerabilities in Actual Execution Environments ◦ Propose Implementation and Maintenance Improvement ◦ Representative Tools (SonarQube, FindSecBug)
DAST	[24-27] [43] [55-56]	◦ Vulnerability Verification in Real-World Environments ◦ Automated CI/CD Environments ◦ Difficult to Trace the Root Cause ◦ Low False Positive Rate ◦ Representative Tools (OWASP ZAP, Burp Suite)

종합적으로 판단해 보았을 때, SAST와 DAST는 각각의 한계를 서로 보완하는 관계로서, SAST와 DAST를 결합한다면 탐지 적용 범위가 크게 향상될 것임을 기대할 수 있다. 따라서 애플리케이션의 전반적인 보안성을 극대화하기 위해서는 SAST와 DAST를 결합하는 HAST 방법론이 필요하다[19]. 이를 통해 OWASP Top 10과 CWE/SANS Top 25 범주의 다양한 취약 항목을 포괄적으로 탐지할 수 있을 것으로 보인다.

III. Novel Approaches for Web Application Vulnerability Detection

3.1. LSTM-based Vulnerability Detection Model

본 논문에서는 다양한 웹 애플리케이션 취약점 탐지를 위한 머신러닝 기법 중 가장 강점을 보이는 LSTM 모델을 제안하고자 한다. LSTM은 HTTP 요청 시퀀스와 같이 순차적·비정형 텍스트 데이터를 처리하여 장기 패턴을 학습하는 것에 강점이 있으므로, NLP(Natural Language Processing) 기법과 결합하여 웹 페이로드 및 파라미터 흐름의 의미를 추출하는 데 유용하다[21][58]. 또한 이전 시점의 정보가 이후 시점의 예측에 중요한 상황에서 장기 의존성을 학습하는 데 능숙하여 장기 의존성이 요구되는 작업에 특히 적합하다. 하지만 계산 복잡성이 높아 학습 시간이 길다는 단점이 존재한다. 이를 최적화하려면 상당한 연산 자원과 효율적인 구현 전략이 필요하다[58].

3.2. Hybrid Application Security Testing (HAST)

기존 SAST와 DAST를 결합한 HAST 접근을 기반으로,

LSTM 기반 사전 탐지 단계를 포함한 확장된 탐지 구조를 제안하고자 한다. 이를 뒷받침하는 선행 연구 결과를 분석하여 SAST와 DAST의 병행 사용 가능성을 확인한 결과[28][39], 보안 테스트 과정의 생산성과 효율성이 향상되었으며, 우선 DAST 도구를 사용하여 SAST의 결과를 수동으로 검증함으로써 오탐을 효과적으로 배제하고, 그다음에 DAST를 수행하여 SAST와 DAST의 결과를 상호 연관시키면 정탐(True Positive) 및 오탐 측면에서 결과의 유효성이 개선됨을 확인하였다[28]. 실무 활용성으로도 보안 소프트웨어 개발 수명 주기(Security Software Development Life Cycle)와 연계가 가능하며, 신규 웹이나 앱뿐만 아니라 운영 중인 서비스에도 적용하여 실제 공격의 가능성 분석 또한 가능함을 확인했다. 이를 통해 SAST와 DAST 및 수동 점검을 통합한 분석이 단독 분석보다 우수한 성과를 보이는 것을 확인 할 수 있었다[28].

또한 OWASP Top 10과 CWE/SANS Top 25 기준으로 웹 애플리케이션 보안 취약점 테스트의 효율성을 연구한 결과로 OWASP ZAP이 가장 효과적인 DAST 도구로 확인되었으며, Yasca가 고위험 취약점 탐지에서 가장 성능이 우수한 SAST 도구로 확인되었다. 아래 Table 6과 Table 7은 SAST, DAST, HAST를 활용한 OWASP Top 10과 CWE/SANS Top 25에 대한 분석 결과이다. 이를 통해 단일 접근법(SAST, DAST)보다 통합 접근법인 HAST가 더 다양하고, 중요한 취약점을 테스트할 수 있다는 사실을 확인할 수 있었다[28].

IV. Proposed Detection Model and Architecture

본 연구는 웹 취약점을 효과적으로 탐지하기 위해, 선행 연구에 대한 문헌 분석을 검토한 결과를 바탕으로 선행 연구들의 한계인 실시간 탐지 대응 미흡, 일반화 문제, 데이터 셋 제약 등의 한계를 보완하기 위해 Multi-Embedding Stacked LSTM을 이용한 실시간 HTTP 요청 탐지와 SAST, DAST를 결합한 HAST 방법론 검증을 통한 탐지 성능을 극대화시키는 모델을 제안하고자 한다. 제안 모델은 총 여섯 단계로 구성되며, 각 단계는 실무 환경에서의 적용성과 탐지 신뢰도를 동시에 향상시키기 위한 목적을 가진다. 다음의 Fig. 4는 연구에서 제안한 모델 아키텍처를 도식화한 것이다.

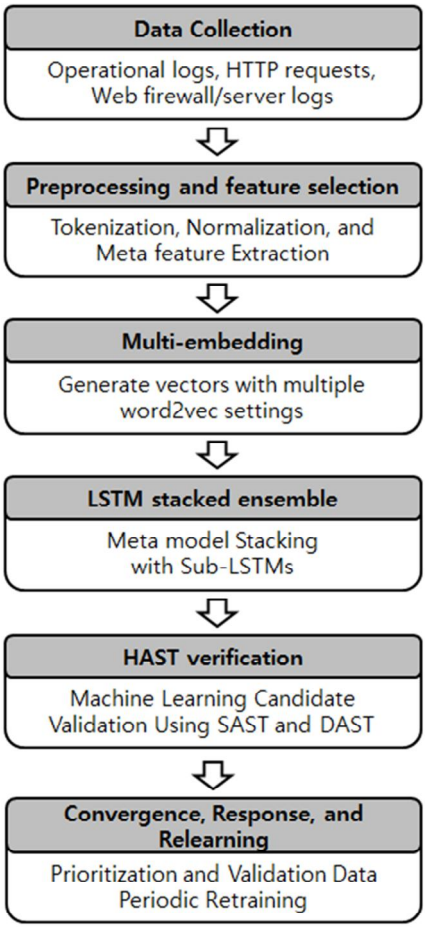

Fig. 4. Proposed Model Architecture

첫 번째 단계는 데이터 수집으로, 운영 환경에서 생성되는 HTTP 요청 로그, 웹서버·애플리케이션 로그, 웹 방화벽 로그 등 다양한 자원으로부터 실시간·배치형으로 데이터를 수집한다. 수집된 로그는 타임스탬프와 소스 IP, 응답 코드 등의 메타데이터를 기준으로 정렬되어 요청 단위의 레코드로 구성되며, 이후 분석을 위한 기본 데이터로 활용된다. 이와 같이 구성된 데이터는 실제 운영 환경의 트래픽 분포와 로그 특성을 반영한 형태로 저장된다.

두 번째 단계는 전처리 및 특징 선택이다. 수집된 원시 로그는 토큰화(URI·파라미터), 정규화(숫자·민감정보 마스킹), 불용어 제거 및 길이 표준화 과정을 거쳐 텍스트 시퀀스와 구조화된 메타 피처(HTTP 메서드, 응답 코드, 세션 지속시간 등)로 변환된다. 이 단계의 출력은 모델 입력에 적합한시퀀스 표현과 정규화된 수치 피처이며, 제안 파이프라인은 도메인 특화 토큰화 규칙과 메타피처를 표준화하여 LSTM 기반 분류기의 입력 품질을 보장한다는 점이 핵심이다.

세 번째 단계는 다중 임베딩으로, 전처리된 토큰 시퀀스는 여러 설정의 단어(토큰)를 실수 벡터로 바꿔서(임베딩) '의미/문맥상의 유사성'을 수치로 표현하게 해주는 방법인 word2vec 또는 사전학습 임베딩을 통해 서로 다른 벡터

Table 6. OWASP Top 10 (number of web applications)

Vulnerable Items	SAST	DAST	SAST + DAST
A01:2021 Broken Access Control	0	75	0
A02:2021 Cryptographic Failures	14	3	1
A03:2021 Injection	7	0	68
A04:2021 Insecure Design	3	22	0
A05:2021 Security Misconfiguration	0	2	73
A06:2021 Vulnerable and Outdated Components	1	58	0
A07:2021 Identification and Authentication Failures	16	0	0
A08:2021 Software and Data Integrity Failures	1	12	0
A09:2021 Security Logging and Monitoring Failures	0	0	0
A10:2021 Server-Side Request Forgery (SSRF)	3	2	0

Table 7. CWE/SANS Top 25 (number of web applications)

Vulnerable Items	SAST	DAST	SAST + DAST
CWE-787: Out-of-bounds Write	0	0	0
CWE-79: Cross-site Scripting	36	0	38
CWE-89: SQL Injection	17	0	56
CWE-416: Use After Free	0	0	0
CWE-78: OS Command Injection	0	2	0
CWE-20: Improper Input Validation	17	0	0
CWE-125: Out-of-bounds Read	0	0	0
CWE-22: Path Traversal	1	27	45
CWE-352: CSRF	0	65	0
CWE-434: Unrestricted Upload of File with Dangerous Type	12	0	0
CWE-862: Missing Authorization	2	40	4
CWE-476: NULL Pointer Dereference	0	0	0
CWE-287: Improper Authentication	70	0	0
CWE-190: Integer Overflow or Wraparound	0	0	0
CWE-502: Deserialization of Untrusted Data	0	0	0
CWE-77: Command Injection	13	0	0
CWE-119: Improper Restriction of Operations within the Bounds of a Memory Buffer	1	18	1
CWE-798: Use of Hard-coded Credentials	74	0	0
CWE-918: Server-Side Request Forgery (SSRF)	4	2	0
CWE-306: Missing Authentication for Critical Function	34	0	0
CWE-362: Race Condition (Concurrent Execution using Shared Resource)	11	0	0
CWE-269: Improper Privilege Management	2	0	0
CWE-94: Code Injection	4	3	0
CWE-863: Incorrect Authorization	0	0	0
CWE-276: Incorrect Default Permissions	0	0	0

표현들로 변환된다. 각 임베딩은 윈도우 크기나 차원수 등 하이퍼파라미터가 달라 서로 다른 문맥적 관점을 제공하며, 출력은 서브 모델별로 입력될 다중 임베딩 시퀀스이다. 제안성은 여기서 드러나는데, 단일 임베딩에 의존하지 않고 표현 다양성을 확보함으로써 변종 공격이나 Zero-day 공격, 희소 패턴에 대한 견고성을 높이려는 점이 본 연구의 기여다.

네 번째 단계는 LSTM 스택드 앙상블로, 각 임베딩을 입력으로 받는 복수의 서브 LSTM이 병렬로 시퀀스 특징을 학습하고 서브 예측을 생성한다. 이후 이들 요약된 예측은 얕은 메타 신경망에 의해 종합되어 최종 악성/정상 확률을 출력한다. 이 단계의 출력은 후보 악성 요청 목록과 서브 모델별 신뢰도 지표이며, 제안 모델은 스태킹을 통해 단일 모델보다 높은 분류 성능과 낮은 오탐률을 달성하도록 설계하였다.

다섯 번째 단계는 HAST 검증 루프이다. LSTM 앙상블이 생성한 후보는 자동으로 SAST 및 DAST 워크플로우로 전달되어 코드 수준의 취약 후보 검증 및 런타임 재현 검증을 수행한다. 이 단계의 출력은 취약성 증거(코드 라인, 유효 페이로드, 재현 결과) 또는 반증 결과이며, 머신러닝 신호를 단순 경고에서 증거 기반의 판정으로 전환해 오탐을 줄이는 점이 본 제안의 핵심 실무적 장점이다.

마지막 단계는 상관·융합 엔진 및 대응·재학습 단계로, LSTM 예측과 SAST/DAST의 증거를 시간과 URI·파라미

터를 기준으로 매칭하고 가중치 기반의 융합 점수를 산출하여 우선 순위화된 취약점 목록을 제공한다. 이 단계에서는 자동 알림·차단 정책이 적용되며, 검증된 사례는 재학습 데이터로 저장되어 주기적으로 모델을 재학습시키게 된다. 제안 파이프라인은 이 피드백 루프를 통해 지속적으로 성능을 개선하며 운영 환경에서의 장기적인 일반화 능력과 실시간 대응력을 확보한다.

종합적으로, 이 모델은 다양한 웹 취약점 유형 탐지에 효과적으로 대응할 수 있으며, 특히 실무 환경에서의 적용성을 극대화하는 가능성을 제시하고 실시간 처리 능력과 제안된 6단계의 사이클을 통해 오탐을 줄이고 변종 공격, Zero-day 공격에 대한 적응성을 높이는 것을 목표로 한다.

이러한 연구 결과를 토대로 기존 연구와 본 연구의 차이점을 요약해 보자면 아래 Table 8과 같다.

Table 8. Comparison with Existing Work

Category	Previous Studies	Present Study
Data Processing Methods	Mostly Static or Log-based Single Analysis	SAST + DAST Integrated Hybrid Structure
Technique Integrity	Single Algorithm Focus	NLP + LSTM + Static/Dynamic Analysis Composite Structure
Detection Range	Specific Attack Type	Full Coverage of OWASP Top 10 available for Mapping
Generalizability	Limited Environments such as WebGoat	Structural Design for Resolving Domain Dependencies (Considering Domain Adaptation)
Real-time Responsiveness	High Detection Accuracy but Speed/False Detection Issues	Structure Designed for Real-time Detection (Enhance Speed and Interpretation)
Design Method	Experiment-driven Performance Verification	Focusing on Literature-based Meta-analysis and Structural Proposals

기존 연구는 정적 로그 기반의 단일 분석이나 특정 알고리즘에 치중하는 경우가 많았으나, 본 연구는 기존 연구들과 달리, 탐지 성능과 실무 적용성을 높였으며, 도메인 적용 기술 및 표준화된 데이터 셋 구축 필요성도 함께 제안한다. 머신러닝 기반 LSTM과 SAST, DAST를 결합한 HAST를 제안함으로써 웹 취약점을 효율적으로 탐지할 수 있는 탐지 모델을 설계하여 제안하였다.

본 연구의 주요 기여는 다음과 같다. 첫째, 기존의 정적 특징 기반 또는 단일 분석 기법 중심의 연구와 달리, Multi-Embedding 기반의 Stacked LSTM을 활용하여 HTTP 요청 시퀀스를 실시간으로 분석하는 탐지 구조를 제안하였다. 이를 통해 웹 요청의 순차적 패턴을 활용한 탐지가 가능해져, 기존 문자열·토큰 단위 분석보다 다양한 공격 양상을 포착할 수 있다. 둘째, SAST와 DAST를 결합한 HAST 아키텍처를 LSTM 기반 모델과 통합하여, 정적 분석·동적 분석·머신러닝 기반 탐지를 하나의 파이프라인에서 수행할 수 있는 구조적 기반을 제시하였다. 이는 기존 연구들이 개별 기법 중심으로 접근한 것과 비교할 때, 개발 단계부터 운영 단계까지 전 주기적 보안 검증이 가능하다는 점에서 차별성을 가진다. 셋째, 제안한 아키텍처는 CI/CD 환경과 연계될 수 있도록 설계됨으로써 실제 DevSecOps 프로세스에서 자동화된 보안 검증을 수행할 수 있는 실무적 확장 가능성을 제공한다. 이러한 점에서 본 연구는 기존의 웹 취약점 탐지 연구가 모델 정확도 중심으로 제한되었던 한계를 넘어, 실시간 탐지와 보안 자동화를 동시에 고려한 새로운 접근을 제시한다는 의의를 갖는다.

V. Conclusions

본 연구는 웹 애플리케이션 보안의 핵심 과제인 OWASP Top 10과 CWE/SANS Top 25를 기반으로 하는 웹 취약점을 효과적으로 탐지하기 위해 다양한 머신러닝 기법을 분석하고, 실제 웹 서비스 환경에서의 적용 가능성을 고려하여 LSTM 기반 머신러닝과 SAST와 DAST를 결합한 HAST 보안 아키텍처를 제안하였다. 이는 SAST, DAST 그리고 머신러닝 기반 탐지를 하나의 파이프라인에서 수행하도록 하는 구조적 기반을 제시하였다는 점에서 의의를 가진다. 제안한 아키텍처는 기존 연구들이 머신러닝 기법과 개별 보안테스트 방법론 중심으로 다루어 왔던 한계를 분석하고 이를 보완할 수 있는 통합적 접근 가능성과 연구 방향성을 제시한다는 점에서 의미를 가진다. 이를 통해 빠르게 변화하는 웹 애플리케이션 환경과 새로운 공격 양상에 대응할 수 있도록, 웹 취약점 탐지를 보다 유연하고 확장 가능한 관점에서 접근할 수 있는 가능성을 모색하였다. 다만, 본 연구는 새로운 아키텍처 설계를 제안하는 탐색적 연구로서 실제 구현 기반의 정량적 실험과 성능 평가는 수행하지 못하였다는 한계를 가진다. 또한 설계한 LSTM 기반의 다중 임베딩 구조와 SAST·DAST 연계 파이프라인은 실시간 탐지 환경에서 높은 연산 비용을 요구할 가능성이 있다. 향후 연구에서는 본 연구에서 제시한 설계를 바탕으로 실증 연구를 수행하여, 지연 시간, 처리량 및 시스템 자원 소모에 대한 검증이 필요할 것이다. 이

와 같은 한계에도 불구하고, 본 연구에서 제안한 통합적 접근은 SAST 및 DAST를 결합한 HAST와 머신러닝 기반 모델을 통해 빠르게 변화하는 웹 애플리케이션 환경에서 새롭게 등장하는 다양한 취약점 유형과 공격 양상을 단계적으로 수용할 수 있는 구조적 유연성을 설계 차원에서 제시하였으며, 이는 향후 웹 취약점 탐지 연구 및 대응 전략의 확장을 위한 기초 연구로 활용될 수 있을 것이다.

ACKNOWLEDGEMENT

This work was supported by the Institute of Information & Communications Technology Planning & Evaluation(IITP)-Innovative Human Resource Development for Local Intellectualization program grant funded by the Korea government(MSIT)(IITP-2026-RS-2022-00156334)

REFERENCES

[1] S. Woo, J. Yoo, “Analysis and Implications of Major Web Application Security Vulnerabilities,” Proceedings of the 2023 Korea Software Congress, pp. 1112-1114, Seoul, Republic of Korea, December 2023.

[2] L. Hu, J. Chang, Z. Chen, and B. Hou, "Web application vulnerability detection method based on machine learning," Journal of Physics: Conference Series, vol. 1827, no. 1, pp. 012061, January 2021. DOI:10.1088/1742-6596/1827/1/012061.

[3] K. Lee, Y. Jung, "Research on Cyber Threat Response Methods Using Machine Learning Technology," Journal of the Korea Academia-Industrial Cooperation Society, vol. 24, no. 10, pp. 829-835, Oct. 2023. DOI: 10.5762/KAIS.2023.24.10.829

[4] H. Ryu, G. Kim, "A Study on Detection Method of Web Attack Using Machine Learning," The Journal of Korean Institute of Communications and Information Sciences, vol. 45, no. 9, pp. 1642-1650, Sep. 2020. DOI: 10.7840/kics.2020.45.9.1642.

[5] E. Lim, “Feature Design Strategy for Web Attack Detection Based on Machine Learning, Master’s Thesis, Graduate School of Information and Communication”, Ajou University, Feb. 2021.

[6] L. Dencheva, "Comparative analysis of Static application security testing (SAST) and Dynamic application security testing (DAST) by using open-source web application penetration testing tools," M.S. thesis, Sch. of Computing, National College of Ireland, Dublin, Ireland, Aug. 2022.

[7] S. Yoon, "Social understanding of security vulnerability: crafting legal & technological strategy," Korea University Graduate School of Information Security, xi, 402 p., 2021.

[8] Web Application, Korea Information and Communications Technology Association Information and Communications Terminology Dictionary. https://terms.tta.or.kr/dictionary/dictionaryView.do?word_seq=097010-1

[9] Security Vulnerability, Korea Information and Communications Technology Association Dictionary of Information and Communications Terminology. https://terms.tta.or.kr/dictionary/dictionaryView.do?word_seq=092901-1

[10] J. Ahn, E. Lee, B. Chang, "A Study on Standard List of Security Weaknesses for Secure SW Development," Journal of the Korea Institute of Information Security & Cryptology, vol. 25, no. 1, pp. 7-18, February 2015.

[11] D. Pandya, N. J. Patel, "OWASP Top 10 Vulnerability Analyses in Government Websites," International Journal of Enterprise Computing and Business Systems, vol. 6, no. 1, pp. 1-7, January-June 2016. ISSN 2230-8849.

[12] OWASP Top 10, https://owasp.org/www-project-top-ten/

[13] J. Seong, H. Lee, I. Ko, K. Kim, "A Study on Web Vulnerability Assessment and Prioritization of Measures by Vulnerabilities," Journal of Convergence Security, vol. 18, no. 3, pp. 38-44, September 2018.

[14] 2024 CWE/SANS Top 25 Most Dangerous Software Errors, http://cwe.mitre.org/top25/

[15] IMPERVA: Application Security Threats, https://www.imperva.com/cyber-threat-index#application-security-threats

[16] S. Qadir, E. Waheed, A. Khanum, S. Jehan, "Comparative evaluation of approaches & tools for effective security testing of Web applications," PeerJ Computer Science, vol. 11, e2821, pp. 1-42, April 2025. DOI:10.7717/peerj-cs.2821.

[17] Korea Internet & Security Agency: "Number of Infringement Incidents", https://www.index.go.kr/unity/potal/main/EachDtlPageDetail.do?idx_cd=1363

[18] T. Micro, "Turning the Tide: Trend Micro Security Predictions for 2021," T. Micro, 2020. https://www.trendmicro.com/vinfo/us/security/research-and-analysis/threat-reports/roundup/trend-micro-security-predictions-for-2021

[19] S. Choi, M. Jang, M. Kim, "A Study on AI Algorithms to Improve Precision Rate in a Managed Security Service, " The Transactions of the Korean Institute of Electrical Engineers, vol. 69, no. 7, pp. 1046-1052, July 2020. DOI: 10.5370/KIEE.2020.69.7.1046

[20] H. Jang, “A Design and Implementation of Deep Learning-Based Intrusion Detection System for Web Applications, Ph.D. dissertation, Graduate School of Technology Management”, Hoseo University, Feb. 2019.

[21] J. Woo, K. Ko, "Development of an AI-Based Security Vulnerability Detection System: Addressing the Limitations of Open-Source Dynamic Analysis Tools," Journal of the Korea

Institute of Information and Communication Engineering and Technology, vol. 17, no. 6, pp. 565-575, Dec. 2024. DOI:10.17661/jkiiect.2024.17.6.565

[22] S. K. Saurabh, D. Kumar, "Model to Reduce DevOps Pipeline Execution Time Using SAST," Research Square (preprint), pp. 1-15, Jul. 2023. DOI: 10.21203/rs.3.rs-2919183/v1

[23] S. M. Elhag, "Study of Open-source Software Adoption Strategy in E-government: Madinah Development Authority," International Journal of Information Technology and Computer Science (IJITCS), vol. 17, no. 1, pp. 16-25, Feb. 2025. DOI: 10.5815/ijitcs.2025.01.02.

[24] T. Alam, A. A. Alshahrani, Y. Alharthi, "Benchmarking OWASP's ZAP V2.12.0 and V2.13.0 for Web Application Security Testing Framework," in Proceedings of the 2023 4th International Conference on Computing and Information Technology (ICCIT), Tabuk, Saudi Arabia, Oct. 2023, pp. 1-6. DOI: 10.1109/ICCIT60884.2023.10483082

[25] V. Nutalapati, "Automated Security Testing for Mobile Apps: Tools, Techniques, and Best Practices," International Research Journal of Engineering & Applied Sciences (IRJEAS), vol. 11, no. 1, pp. 26-31, Jan.-Mar. 2023. DOI: 10.55083/irjeas.2023.v11i01004

[26] K. S. Solanki, H. B. Soni, "Survey on Web Application Security Testing Methods," International Journal of Advanced Computer Science and Applications (IJACSA), vol. 11, no. 5, pp. 646-652, 2020. DOI: 10.14569/IJACSA.2020.0110582

[27] D. B. Cruz, J. R. Almeida, J. L. Oliveira, "Open Source Solutions for Vulnerability Assessment: A Comparative Analysis," IEEE Access, vol. 11, pp. 100234-100249, Sept. 2023, DOI: 10.1109/ACCESS.2023.3315595

[28] R. A. Correa, J. R. Bermejo Higuera, J. Bermejo, J. A. Sicilia Montalvo, M. Sańchez, Á. A. Magreñań,"Hybrid Security Assessment Methodology for Web Applications," Computer Modeling in Engineering & Sciences, vol. 126, no. 1, pp. 89-122, 2021. DOI: 10.32604/cmes.2021.010700

[29] S. Qadir, E. Waheed, A. Khanum, S. Jehan, "Comparative evaluation of approaches & tools for effective security testing of Web applications," PeerJ Computer Science, vol. 11, e2821, Apr. 2025. DOI: 10.7717/peerj-cs.2821

[30] TTA Information and Communication Terminology Dictionary: Supervised learning, https://terms.tta.or.kr/dictionary/dictionaryView.do?word_seq=184207-1

[31] TTA Information and Communication Terminology Dictionary: Unsupervised learning, https://terms.tta.or.kr/dictionary/dictionaryView.do?word_seq=184205-1

[32] W. Lee, M. Lee, D. Seo, "Application of Machine Learning Techniques for the Classification of Source Code Vulnerability," Journal of The Korea Institute of Information Security & Cryptology, vol. 30, no. 4, pp. 735-743, Aug. 2020, DOI: 10.13089/JKIISC.2020.30.4.735

[33] C. Han, S. Yun, M. Han, I. Lee, "Machine Learning-Based Malicious URL Detection Technique," Journal of The Korea Institute of Information Security & Cryptology, vol. 32, no. 3, pp. 555-564, Jun. 2022, DOI: 10.13089/JKIISC.2022.32.3.555

[34] P. Shahrivar, "Detection of Vulnerability Scanning Attacks using Machine Learning: Application Layer Intrusion Detection and Prevention by Combining Machine Learning and AppSensor Concepts", MSc Thesis, KTH Royal Institute of Technology, Stockholm, Sweden, 2021.

[35] I. M. Marçal, Evaluation of Static Analysis Tools in Detecting OWASP Top 10 Vulnerabilities, Master's Dissertation, Dept. of Informatics Engineering, Faculty of Sciences and Technology, University of Coimbra, Coimbra, Portugal, Sep. 2024

[36] M. Y. Tanko, A. B. Md Sultan, M. H. Osman, H. Zulzalil, "An Approach for Vulnerability Detection in Web Applications Using Graph Neural Networks and Transformers," Journal of Theoretical and Applied Information Technology, vol. 103, no. 1, pp. 257-265, Jan. 2025. ISSN: 1992-8645.

[37] M. Huang, J. Chen, "Web Application Security Education Platform Based on OWASP API Security Project," in Proceedings of the 2020 2nd International Conference on Information Technology and Computer Application (ITCA), Guangzhou, China, Dec. 2020, pp. 1-6. DOI: 10.1109/ITCA52113.2020.00009

[38] P. A. Sarpong, L. S. Larbi, D. P. Korsah, I. B. Abdulai, R. Amankwah, A. Amponsah, "Performance Evaluation of Open Source Web Application Vulnerability Scanners based on OWASP Benchmark," International Journal of Computer Applications, vol. 174, no. 18, pp. 15-22, Feb. 2021. DOI: 10.5120/ijca2021921171

[39] M. E. Mahmoud, M. T. Thirugnanasambandam, and S. Shukla, "A Comprehensive Survey on Vulnerability Detection and Mitigation Techniques in Web Applications," arXiv preprint arXiv:2203.09938, pp. 1-15, March 2022.

[40] H. Lee, K. Kim, "Novelty Detection on Web-server Log Dataset," Journal of the Korea Institute of Information and Communication Engineering, vol. 23, no. 10, pp. 1311-1319, Oct. 2019. DOI: 10.6109/jkiice.2019.23.10.1311

[41] N. M. Min, V. Visoottiviseth, S. Teerakanok, N. Yamai, OWASP IoT Top 10 based Attack Dataset for Machine Learning, in Proceedings of the International Conference on Advanced Communications Technology (ICACT 2022), Seoul, Republic of Korea, Feb. 13-16, 2022, pp. 317-322. ISBN: 979-11-88428-08-3

[42] V. Bril, "Automation of Remediation of Configuration Vulnerabilities Reported by the DAST Scanning Procedure," MSc Thesis, National College of Ireland, Jan. 2023.

[43] G. Betarte, R. Martínez, Á. Pardo, "Web Application Attacks Detection Using Machine Learning Techniques," in Proceedings of the 2018 IEEE Latin American Computing Conference (CLEI),

São Paulo, Brazil, Oct. 2018, pp. 1-8. DOI: 10.1109/CLEI.2018.00012

[44] A. Almorjan, M. Basheri, M. Almasre, "Large Language Models for Synthetic Dataset Generation of Cybersecurity Indicators of Compromise," Sensors, vol. 25, no. 9, p. 2825, Apr. 2025. DOI: 10.3390/s25092825

[45] ESET Latinoamérica, "ESET Security Report Latinoamérica 2016," WeLiveSecurity. https://www.welivesecurity.com/la-es/

[46] S. M. Ghaffarian, H. R. Shahriari. "Software vulnerability analysis and discovery using machine learning and data-mining techniques," ACM Computing Surveys, vol. 50, no,4, pp. 1-36. 2017. DOI 10.1145/3092566

[47] P. Nunes, I. Medeiros, J. Fonseca, N. Neves, M. Correia, et al. "An empirical study on combining diverse static analysis tools for web security vulnerabilities based on development scenarios," Computing, vol. 101, no. 2, pp. 161-185, 2019. DOI 10.1007/s00607-018-0664-z

[48] C. Páez, M. Michel, "Application Security Testing Tools Study and Proposal," Master Universitario en Seguridad de las Tecnologías de la Información y de las Comunicaciones, pp. 1-53, December 2020.

[49] G. Bennett, T. Hall, E. Winter, S. Counsell, "Semgrep*: Improving the Limited Performance of Static Application Security Testing (SAST) Tools," Proceedings of the 28th International Conference on Evaluation and Assessment in Software Engineering (EASE 2024), pp. 1-10, Salerno, Italy, Jun. 2024. DOI: 10.1145/3661167.3661262

[50] K. Kuszczyński, M. Walkowski, "Comparative Analysis of Open-Source Tools for Conducting Static Code Analysis," Sensors, vol. 23, no. 18, p. 7978, Sep. 2023. DOI: 10.3390/s23187978

[51] A. Bakhshandeh, A. Keramatfar, A. Norouzi, M. M. Chekidehkhoun, "Using ChatGPT as a Static Application Security Testing Tool," ISeCure: The ISC International Journal of Information Security, vol. 15, no. 3, pp. 1-8, 2023. ISSN: 2008-2045

[52] R. A. Gowda, N. Heffernan, "Performance Evaluation of Automated Web Vulnerability Scanners for Cross Platforms - Red Teaming," National College of Ireland, 22, 2023.

[53] J. Li, "Vulnerabilities Mapping based on OWASP-SANS: A Survey for Static Application Security Testing (SAST)," Annals of Emerging Technologies in Computing (AETiC), vol. 4, no. 3, pp. 1-8, July 2020. DOI: 10.33166/AETiC.2020.03.001

[54] K. Li, S. Chen, L. Fan, R. Feng, H. Liu, C. Liu, Y. Liu, Y. Chen, "Comparison and Evaluation on Static Application Security Testing (SAST) Tools for Java," Proceedings of the 31st ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering (ESEC/FSE '23), pp. 1-13, San Francisco, USA, December 2023. DOI: 10.1145/3611643.3616262

[55] S. S. Kalyankar, P. Bhutekar, "Automating Web Application Security Testing in Continuous Integration and Deployment Environments", International Journal for Research in Applied Science & Engineering Technology (IJRASET), vol. 11, no. 7, pp. 3172-3176, Jul. 2023. DOI: 10.22214/ijraset.2023.56440

[56] G. Bhadreshsinh, K. Gohil Rishi, A. Pathak Axaykumar Patel, "Federated Network Security Administration Framework," International Journal of Computer Science and Mobile Computing (IJCSMC), vol. 2, no. 3, pp. 52-56, Mar. 2013.

[57] P. A. Sarpong, L. S. Larbi, D. P. Korsah, I. B. Abdulai, R. Amankwah, A. Amponsah, "Performance Evaluation of Open Source Web Application Vulnerability Scanners based on OWASP Benchmark," International Journal of Computer Applications, vol. 174, no. 18, pp. 1-8, February 2021.

[58] S. Hochreiter, J. Schmidhuber, "Long Short-Term Memory," Neural Computation, vol. 9, no. 8, pp. 1735-1780, Dec. 1997. DOI: 10.1162/neco.1997.9.8.1735

Authors

Ha Young Kim received her bachelor's degree in Computer Information and Security in 2019. She is currently pursuing the M.S. degree in Information Security at Pai Chai University.

Her current research interests include information security, privacy protection, artificial intelligence, and malware.

Seong-Cho Hong received the B.A. degree in Psychology in 2014, M.A. in Criminology in 2018 and Ph.D. in Legal Psychology. He joined the Smart ICT Convergence Human Resource Development Center at Pai Chai University, Daejeon, South Korea, as a Research Professor. His current research interests include security, artificial intelligence, criminal behavior, theoretical framework and convergence research.

Ah Reum Kang received the M.S. and Ph.D. degrees in information security from Korea University, South Korea, in 2012 and 2016. She is a professor in the Department of Information Security at Pai Chai University in Daejeon, South Korea. Her current research interests include security, artificial intelligence, malware, medical data analysis, and online game security.