BULLETIN of Cherkasy State Technological University

Journal homepage: https://bulletin-chstu.com.ua/en

Vol. 30 No. 3. 2025

UDC 004.056.5:004.8
DOI: 10.62660/bcstu/3.2025.47

Possibilities and limitations of artificial intelligence in vulnerability testing: Effectiveness of the approach as opposed to traditional pentesting

Igor Rudenko*
Master
Yaroslav Mudryi National Law University
61024, 77 Hryhorii Skovoroda Str., Kharkiv, Ukraine
https://orcid.org/0009-0008-3582-3951

Abstract. The research relevance is determined by the growing threats in cybersecurity, which require improved methods of detecting vulnerabilities in software, with the help of the latest technologies such as artificial intelligence (AI). The study aimed to determine the effectiveness of AI for vulnerability testing as an alternative to traditional security testing methods, particularly pentesting. To achieve this goal, machine learning algorithms were analysed, a hybrid model for vulnerability detection was developed, and the use of intrusion detection and prevention systems, automated approaches to software testing, and application interface security was addressed. The study determined that decision trees quickly classify traffic but overlearn; support vector machines accurately analyse logs but are sensitive to settings; naive Bayesian classifiers effectively filter messages but are limited by assumptions; neural networks and deep learning detect complex threats but require a lot of data; the k-nearest neighbours' algorithm is suitable for small systems but slow; and random forests are accurate in code analysis but less interpretable. AI is fast and scalable but limited in understanding context and responding to new threats. The results showed that pentesting prevailed in detecting complex vulnerabilities, and the hybrid machine learning model achieved 93% accuracy and 100% prediction accuracy but missed 17% of vulnerabilities. For their part, cybersecurity technologies such as application interface testing, bug bounty programmes, security integration into development, intrusion detection systems, end-to-end encryption, IoT security and phishing are also effective, but need to be adapted to new threats. The results of the study can be used by cybersecurity companies, software developers, and security engineers to improve vulnerability testing and cyber threat protection processes, including with the help of AI-based tools

Keywords: machine learning algorithms; hybrid analysis models; intrusion detection and prevention systems; software verification automation; application interface protection; attack simulation scenarios

INTRODUCTION

Given the rapid development of information technology, the issue of cybersecurity is becoming particularly relevant, as the growing number of digital services and the amount of personal and confidential data transmitted over the network require effective detection of software vulnerabilities. Modern approaches to security testing are based on the principles of vulnerability detection, penetration testing, as well as the concepts of Artificial Intelligence (AI) and Machine Learning (ML), which can be used for creation of automated systems for analysing and responding to threats. The research relevance is determined by the fact that traditional methods, in particular manual pentesting, have several limitations, such as significant time and resource costs, dependence on the human factor, lack of scalability and low efficiency in identifying the latest threats.

Article's History: Received: 02.05.2025; Revised: 28.07.2025; Accepted: 15.09.2025.

Suggested Citation:
Rudenko, I. (2025). Possibilities and limitations of artificial intelligence in vulnerability testing: Effectiveness of the approach as opposed to traditional pentesting. *Bulletin of Cherkasy State Technological University*, 30(3), 47-60. doi: 10.62660/bcstu/3.2025.47.

*Corresponding author

In turn, AI technologies and ML algorithms have the potential to improve detection accuracy, automate security analysis processes, reduce the number of false positives, and adapt more quickly to new attacks. However, these systems also have several disadvantages, including limited ability to detect fundamentally new threats, lack of model explanation, dependence on the quality of training samples, and the risk of generating false conclusions. In this regard, there is a need for a comprehensive study of the possibilities and limitations of using AI for vulnerability testing, assessing the effectiveness of different approaches and developing recommendations for their integration with classical cyber defence methods.

For instance, F. Shvets *et al.* (2024) investigated the potential of AI for detecting deviations and anomalies in user behaviour using analytical tools and automated control systems, which is relevant for cybersecurity tasks, particularly the detection of potentially malicious activity. A. Kudriashov (2024) analysed the integration of AI into the security architecture of 5G and 6G mobile networks, emphasising the need to implement the concept of zero trust and post-quantum cryptography to counter modern cyber threats in the context of increasing network technology complexity. In addition, O. Trofymenko *et al.* (2024) analysed the application of AI for military network cybersecurity, focusing on the use of intelligent agents, ML and deep learning (DL) for real-time threat detection, analysis and mitigation, highlighting the potential of AI in proactive protection of critical infrastructures.

On the other hand, M. Nasir & J. Pomeroy 2025) examined the transformative impact of AI on ethical hacking, including the increased efficiency, accuracy, and scalability of vulnerability assessment and pentesting. The study emphasised the ability of AI to automate vulnerability detection, predict attack vectors, conduct exploit simulations, and improve post-exploitation analysis, which significantly enhances cyber defence capabilities. The results of A.M. Akinyemi & S. Sims (2025) demonstrated that the use of AI in vulnerability management increases the accuracy and speed of threat detection, reducing false positives and optimising resource use. The study emphasised the effectiveness of a hybrid approach that combines human expertise with AI. In turn, the study by S.A. Teo (2025) proposed the concept of "algorithmic vulnerability" as a new theoretical framework for understanding the risks arising from AI, which generates unpredictable and dynamic threats. It substantiates the need for a multi-level resilience strategy to overcome both traditional and new forms of vulnerabilities in the digital age. At the same time, E. Cruz (2025) considered human vulnerability as a potential defence mechanism in the context of AI development, in particular its ability to analyse personal data and interpret mental processes. The study emphasised that the limited and opaque nature of the human inner world can serve as a barrier against excessive AI intervention and help preserve individual autonomy in the context of digital surveillance.

N. Wang *et al.* (2025) found that AI has a positive impact on the innovation of sustainable business models, through strategic flexibility, and the dynamism of the environment enhances this effect. The results emphasised the importance of adaptability and the use of AI to effectively respond to changes and opportunities in a volatile market. A study by H.M. Almheiri *et al.* (2024) highlighted that AI has a positive impact on dynamic organisational capabilities, creativity, and overall performance of public entities, with dynamic capabilities and creativity mediating this relationship. This underscores the potential of AI to enhance decision-making and productivity in the public sector through targeted management of internal resources. Additionally, G. Capelli *et al.* (2023) highlighted the ethical aspects and reliability of AI use. The study reviewed six main topics: reliability of AI systems, respect for privacy, use of complete and representative data, transparency and uncertainty of AI, equity in access to technology, and the role of technology in equality of opportunity.

The study aimed to determine how effective AI can be in detecting vulnerabilities compared to traditional security testing methods, which are not sufficiently covered in the reviewed works. The tasks included a comprehensive analysis of the capabilities and limitations of AI in vulnerability detection, as well as a study of the current challenges, advantages, and limitations of traditional pentesting and related cybersecurity technologies.

MATERIALS AND METHODS

The research was conducted as a theoretical and empirical study, combining theoretical literature analysis and empirical modelling to assess the effectiveness of AI in vulnerability testing compared to traditional pentesting. The analysis was based on open sources, including reports and scientific articles, as well as implementation of testing through software modelling. For the theoretical part, data on ML algorithms such as Decision Trees (DT), Support Vector Machines (SVM), Naive Bayes Classifiers (NBC), Neural Networks (NN), DL, k-Nearest Neighbours (k-NN), and Random Forest (RF) were used (Putri *et al.*, 2024). The choice of these algorithms was based on their popularity in cybersecurity, ability to process various types of data (traffic, logs, code), and adaptability to hybrid systems (Nguyen *et al.*, 2024).

The empirical part included the development of a custom Python program using pandas (for data processing), numpy (for number generation), sklearn (for model creation and evaluation), and matplotlib (for visualisation) to create a synthetic dataset of 50 HTTP requests simulating vulnerabilities such as Structured Query Language (SQL) injection and Cross-Site Scripting (XSS) with a clear feature dependency and minimal noise. The simulation was implemented through a hybrid model combining RF and SVM algorithms (chosen due to the high resilience of RF to overtraining and the

ability of SVM to work effectively with low-dimensional and complex data), with data normalisation and training on balanced classes, and the results were interpreted using charts and statistical metrics (Accuracy, Precision, Recall). To compare AI approaches with pentesting, values based on literature data were used (Pratama *et al.*, 2024; Kumar, 2025; Thool & Brown, 2025).

Automation of the testing of Application Programming Interfaces (APIs), including authentication, authorisation, access restrictions, and detection of excessive data disclosure, was addressed (Neelapu & Pub, 2023). The visualisation of automated testing and examples of tools for detecting excessive data disclosure through APIs displayed the key stages of system-level testing (Pan *et al.*, 2024). The role of bug bounty programmes was also analysed, and examples of the effective use of such programmes in modern conditions were given (Mayoral-Vilches *et al.*, 2025). The study examined the Development, Security, Operations (DevSecOps) methodology, which involves the implementation of security as an integral element of the entire software development cycle. A comparative analysis of Intrusion Detection System (IDS) and Intrusion Prevention System (IPS) was conducted. The evaluation was based on several criteria: purpose, principle of operation, typical deployment point, response to the threat, application examples, and the level of integration with AI tools (Karambelkar, 2025).

Moreover, the study also examined such topical areas of cybersecurity as Phishing 3.0, end-to-end encryption, and challenges related to the security of Internet of Things (IoT) devices. The final part of the study provided a comparative analysis of all the technologies considered, from classical pentesting to modern automated approaches and AI solutions, based on such metrics as the level of automation, the ability to detect both typical and complex logical vulnerabilities, integration compatibility, flexibility in application, human resource requirements, implementation and support costs, and sensitivity to new or complex threats. All visualisations were implemented in the Python programming language using the Visual Studio Code environment, which was used to efficiently structure the results and ensure the integration of the analytical and visual components of the study. The method of data collection involved the use of research results through their analysis and synthesis, and the interpretation of the results was based on statistical analysis and visualisation to ensure an objective assessment of effectiveness.

RESULTS

Analysing the effectiveness of modern approaches to detecting AI-based vulnerabilities

Given the rapid development of digital technologies, effective detection of vulnerabilities in software is critical to cybersecurity. One of the most promising areas in this field is the use of AI methods, in particular ML algorithms (Table 1). They can be used for automating the processes of analysing code, network activity, and user behaviour, which, in turn, can significantly accelerate and scale the detection of potential threats.

Table 1. ML algorithms for software vulnerability detection

Algorithm	Possibilities	Limitations
DT	Easy for interpretation	Propensity to relearn
	Speed of learning	Limited accuracy
SVM	Efficiency with small amounts of data	Heavy scaling
	High precision	Sensitivity to settings
NBC	Speed	Assumption of independence of features, which is often not true
	Easy to implement	
NN	Power in detecting hidden patterns	Requirement for large data sets
	Ability to process big data	Difficulty of interpretation
DL	High precision	High computing costs
	Automatic feature extraction	"Black Box"
k-NN	Easy to use	Low large data amount processing rate
	No training required	Sensitivity to noise
RF	High precision	Less interpretable result
	Resilient to relearning	Slow prediction with a large number of trees

Source: compiled by the author based on A.I. Putri *et al.* (2024)

That is, DT methods are used to quickly classify network traffic and detect potential malicious activity, for example, behavioural analysis to detect attacks. SVMs are effective for detecting anomalies in system logs (detecting SQL injections or XSS attacks). As for NBCs, they are suitable for fast real-time filtering of spam or malicious messages. In turn, NNs are used for complex threats, such as zero-day attacks, for example, analysing logs to detect abnormalities. DL, on the other hand, is used to analyse unstructured data, such as text logs or query sequences, to detect multi-vector attacks. The k-NN algorithm is used in small systems to classify packets or requests, for example, in local networks to determine traffic security.

At the same time, RFs analyse source code or variables to detect vulnerabilities, for example, in code or configuration analysis systems. Thus, the effective use of AI in vulnerability detection often involves not so

much the selection of one best algorithm as a combination of approaches or their integration into larger hybrid systems, considering current security constraints and goals. As shown in Figure 1, the components of an AI-based hybrid system interact with each other to effectively detect vulnerabilities. The positions of the components indicate their relationships and roles in the vulnerability testing process, which simplifies the perception of system architecture. Different methods, such as data collection, vulnerability analysis, and others, work together to improve the results in identifying potential threats.

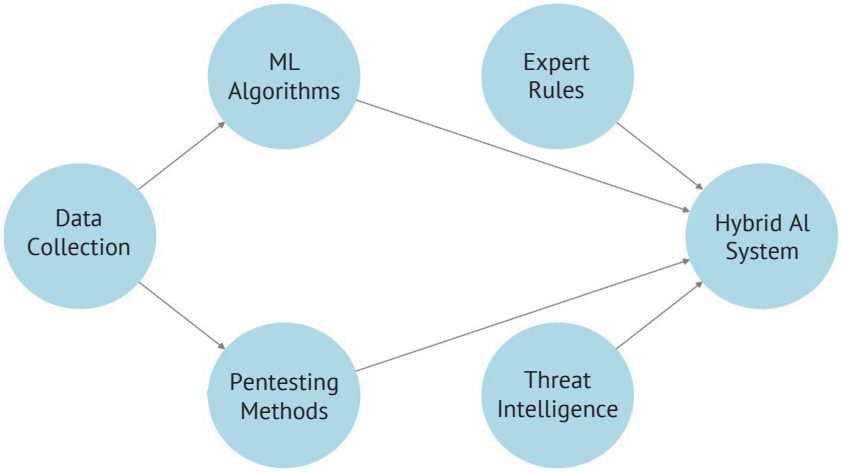

Figure 1. Architecture of an AI-based hybrid system for vulnerability detection
Source: compiled by the author based on H.P. Nguyen *et al.* (2024)

This diagram illustrates the relationships between the main components of a hybrid vulnerability detection system, including data collection methods, ML algorithms, and traditional vulnerability testing approaches. Each component has a specific role in the decision-making process, which improves accuracy and efficiency of diagnosis of system vulnerabilities. However, despite their high technical capabilities, AI methods are not sufficiently capable of deep contextual understanding. In the field of vulnerability testing, this is manifested in the fact that even the most advanced models may not address the specifics of business logic, user roles, or peculiarities of interaction with software. For instance, a certain action may look harmful from a technical point of view but be perfectly acceptable from a business process perspective. This is especially critical in complex or non-standard systems, where the rules for interacting with data depend on the context, which is difficult to formalise in the form of a training dataset. As a result, models can produce false-positive or false-negative results because they analyse mainly formal features rather than the logic of user behaviour or system goals. This limitation requires additional intervention from specialists or the use of hybrid approaches that combine AI with classical analysis methods.

Another major challenge for using AI in vulnerability testing is its limited ability to respond to new, previously unknown types of attacks. ML algorithms work based on the data they have been trained on, and their effectiveness directly depends on the completeness and relevance of this data. If a certain vulnerability or attack scenario is not included in the training set, the model may simply not recognise it as a threat. This is a particularly critical problem in the rapidly changing cyber threat landscape, where new security circumvention techniques, zero-day attacks, and complex multi-vector incidents are constantly emerging. In such situations, AI systems may be powerless if there are no mechanisms for continuous training or integration with external knowledge sources (e.g., threat intelligence). Therefore, the use of AI in cybersecurity requires not only one-time training but also constant updating of models and verification of their relevance.

In general, AI, similarly to any automation tool, is not immune to false decisions, including false positives and false negatives. In the first case, the system can classify legitimate activity as malicious, which leads to unnecessary alarms, overloading security specialists, and blocking the normal work of users. In the second case, the threat will not be detected, which opens a potential opportunity for a successful attack. False negatives are particularly critical, as they create a false sense of security. The reasons for such errors vary from an incomplete training set to overgeneralisation of the model. An excessive number of false positives, in turn, reduces the confidence in the system on the part of professionals and may lead to the ignoring of important alerts. Thus, to really improve the level of cyber defence, it is necessary not only to implement AI solutions but also to constantly monitor their accuracy and combine models with human expertise.

Despite its limitations, AI is a highly useful tool in cybersecurity. It can operate in real time, responding quickly to suspicious events or behavioural anomalies,

which significantly reduces the response time to potential incidents. In addition, AI models can detect complex correlations between events that are difficult or impossible to detect manually, which increases the level of proactive protection. Another advantage is the ability to self-learn modern algorithms can improve their efficiency as new data is accumulated, adapting to changes in the environment. Thanks to this, AI technologies not only automate routine processes but are also able to detect new types of threats before they become widespread.

Comparison of AI approaches and pentesting

Despite the growing role of AI in information security, traditional approaches to vulnerability testing remain relevant and indispensable in many scenarios. Professional pentesting, focused on the practical simulation of the actions of real attackers, remains central in the cyber resilience of organisations (Fig. 2). In contrast to fully automated AI solutions, pentesting can be used for atypical configurations, complex logical errors, and contextual risks that often remain outside the scope of algorithmic analysis.

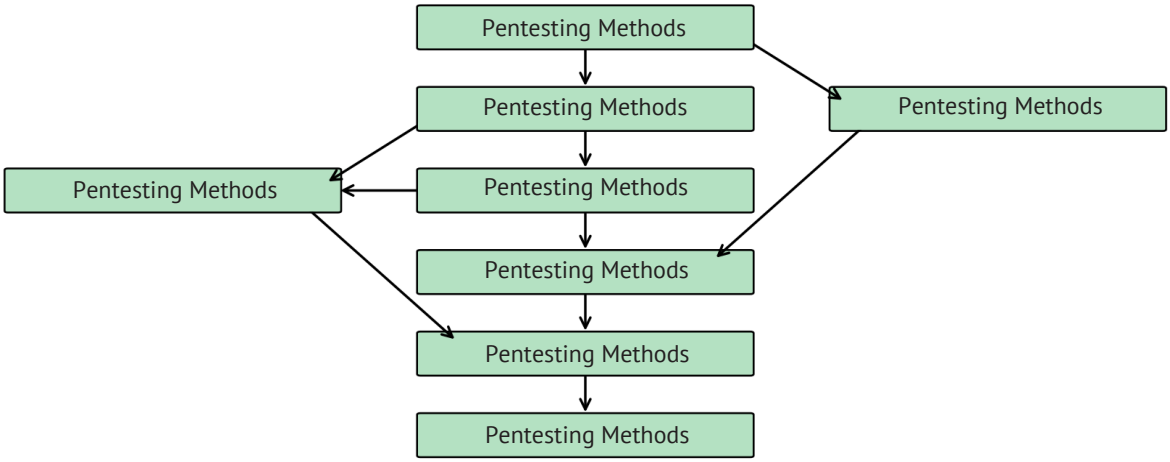

Figure 2. Diagram of the traditional pentesting process

Source: compiled by the author based on D. Pratama *et al.* (2024)

The diagram shows the key stages of penetration testing from initial reconnaissance to final retesting. This visualisation demonstrates both the standard course of pentesting and the flexibility of the process depending on the identified vulnerabilities and the response of the system under test. In general, in 2025, pentesting techniques have significantly expanded to cover more complex attack scenarios that address architectures such as microservices, dynamic Continuous Integration/Continuous Delivery (CI/CD) environments, multi-level authentication mechanisms, and integration with event log monitoring and analysis systems (Thool & Brown, 2025). With the growth of digital infrastructure and the number of threats, pentesting is becoming critical to identifying vulnerabilities and preventing incidents. For a comprehensive system check, black-box, white-box, and grey-box testing approaches are used, as well as the standards of the Open Worldwide Application Security Project and the Open-Source Security Testing Methodology Manual (Kumar, 2025). The focus is on legal and ethical aspects, as well as responsible disclosure of vulnerabilities.

Although AI systems are making significant progress in identifying common vulnerabilities, they cannot completely replace professional pentesters. Human expertise remains indispensable in situations requiring contextual comprehension, creative thinking, and adaptation to non-standard conditions. At the same time, automated solutions are becoming an effective complement to traditional pentesting due to their high analysis speed, ability to process large amounts of data, and autonomous adaptation to new types of threats. Such solutions include static and dynamic code analysis systems with AI elements, automated scanning frameworks, exploit generators, and real-time system behaviour analysis tools. However, their effectiveness directly depends on the context of application, configuration, and depth of the built-in models.

Despite the speed and scalability advantages of AI, its integration into vulnerability testing does not eliminate the need for traditional pentesting, but rather complements it, creating a synergy between automation and human expertise. Modern AI-based tools, such as static and dynamic analysis systems, can be used to quickly process large amounts of data and identify common vulnerabilities with high accuracy. However, their effectiveness depends on the quality of training data and settings, which can limit them in real-world scenarios. In this context, a hybrid approach that combines AI and pentesting becomes the optimal solution, ensuring a balance between the speed of automation and the depth of human analysis. To compare the effectiveness of these approaches in detail, a Python program that implements a hybrid ML model for detecting vulnerabilities in web applications, compares its effectiveness with the results of manual pentesting, and creates a visualisation of the results was created (Fig. 3).

```
data['is_vulnerable'] = is_vulnerable

df = pd.DataFrame(data)

X = df.drop('is_vulnerable', axis=1)
y = df['is_vulnerable']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)

scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)

rf = RandomForestClassifier(n_estimators=100, random_state=42, class_weight='balanced')
svm = SVC(probability=True, C=1.0, random_state=42, class_weight='balanced')
hybrid_model = VotingClassifier(estimators=[('rf', rf), ('svm', svm)], voting='soft')

hybrid_model.fit(X_train_scaled, y_train)

proba = hybrid_model.predict_proba(X_test_scaled)[:, 1]
y_pred = (proba > 0.4).astype(int)
```

Figure 3. Python code snippet of a hybrid AI model for vulnerability testing
Source: compiled by the author

First, the necessary libraries are imported, such as pandas, numpy, sklearn, and matplotlib. A synthetic dataset of 50 records is created that simulates HTTP requests with features such as request length, number of special characters, method type (GET/POST), and the presence of SQL injection or XSS patterns. This set is generated with a clear relationship between the features and vulnerability labels, with minimal noise added. This approach demonstrates the ability of AI to automatically model data, which significantly speeds up the process compared to manual collection of vulnerability examples by a pentester, but highlights limitations due to the potential dependence on data quality.

The next stages of the programme focus on data preparation and model training. The data is normalised and split into training and test samples. A hybrid model that combines RF and SVM is trained with balanced class weights, which efficiently recognises both safe and malicious queries. As a result, the metrics (Accuracy, Precision, Recall) are calculated for the test sample, and the chart compares them with the pentesting results. In this case, the ML model reaches higher values, which illustrates its advantage in automated detection of typical vulnerabilities, as evidenced by the correct classification of a new request (Fig. 4).

```
Accuracy: 0.93
Precision: 1.00
Recall: 0.83
False Positives (FP): 0
False Negatives (FN): 1
Prediction for new request: Safe
Comparison plot saved as 'comparison_plot.png'
```

Figure 4. Evaluation of model performance by metrics
Source: compiled by the author

The first metric (Accuracy) means that the model correctly classified 93% of the queries in the test sample. This indicates a high overall accuracy, but since the classes may be unbalanced, this metric should be considered along with other metrics. Metrics such as Precision and Recall provide a deeper understanding of the model's performance. They show that all the requests that the model labelled as malicious were indeed malicious (no false positives), which is a significant advantage in cybersecurity, as false alarms can waste analysts' time. At the same time, the model missed one malicious request, meaning that 17% of vulnerabilities went undetected. This highlights the limitations of AI: although the model avoids false alarms, it can miss critical vulnerabilities, which can be dangerous in real-world conditions. For better visualisation, a chart that compares ML and pentesting, helping to explore their strengths and weaknesses, was created (Fig. 5).

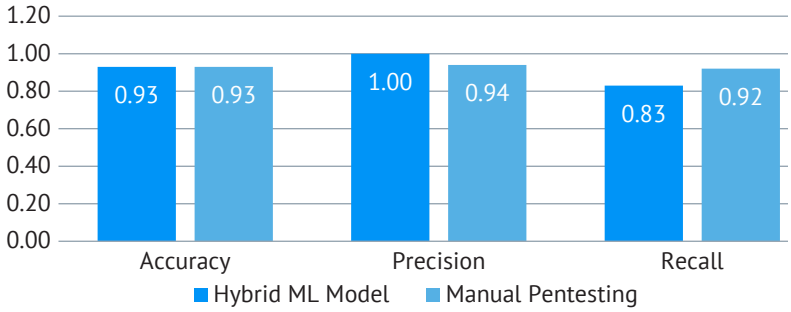

Figure 5. A diagram comparing an ML model and manual pentesting
Source: compiled by the author

Visually, the diagram shows that the ML model outperforms pentesting in Precision (1.00 vs. 0.94), which emphasises its ability to avoid false alarms, which is an advantage in automated testing. However, the pentesting is better in terms of Recall (0.83 vs. 0.92), which illustrates the strength of human analysis – higher sensitivity to vulnerabilities, as the pentester can address the context that the ML model missed. In terms of Accuracy, both approaches have the same value (0.93), which indicates parity in overall accuracy. In other words, the diagram emphasises the balance between the speed and accuracy of AI and the depth of pentesting analysis, suggesting a hybrid approach as the best for vulnerability testing.

When it comes to generating attack scenarios in the context of integrating AI into the pentesting process, AI opens new opportunities for automating this process, enabling the creation of diverse and complex scenarios based on large amounts of data about previous attacks. Thanks to its ability to analyse threat patterns in real time, AI can generate scenarios that mimic the actions of attackers, including attacks on microservices, CI/CD environments, or multi-level authentication systems, which significantly expands the range of testing. However, the effectiveness of such scenarios depends on the quality of the training data and algorithms, as well as human supervision, as AI may not cover unique contextual factors or ethical constraints that a pentester assesses intuitively, which underscores the importance of AI-human integration rather than AI replacing humans.

Prospects and challenges of modern cybersecurity technologies

APIs are one of the most vulnerable and yet most common components of digital ecosystems. Automating API testing can detect common threats, such as insufficient authentication, injection, mismanagement of authorisation, or sensitive information leaks (Fig. 6). Modern tools, including AI-enhanced scanners, can automatically create requests, check server responses, and detect anomalies in data processing.

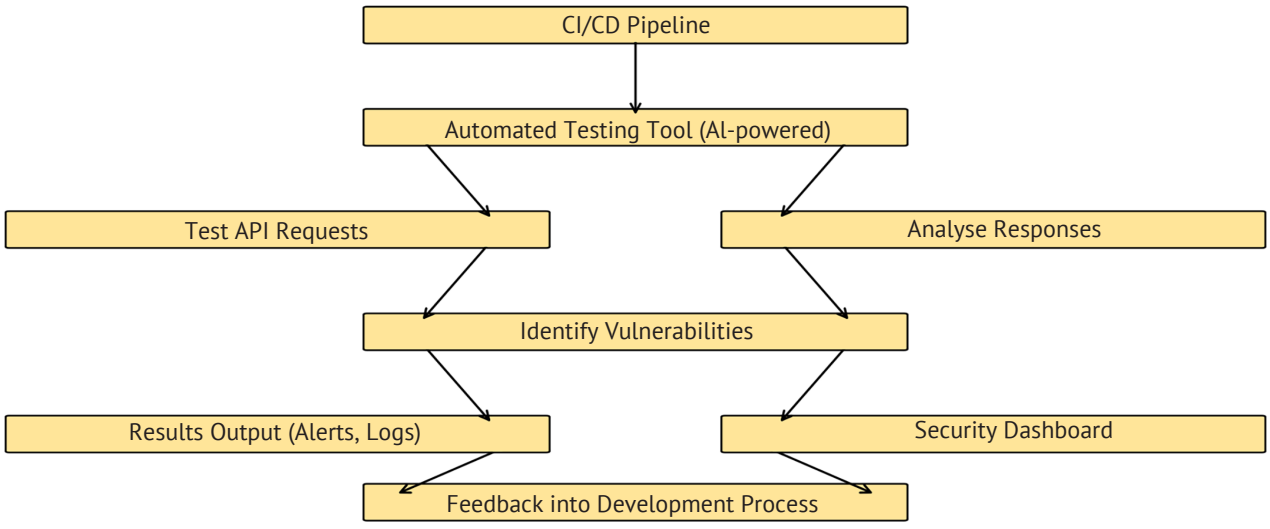

Figure 6. Automated API security testing
Source: created by the author based on M. Neelapu & A. Pub (2023)

This illustration demonstrates a branched logic where several parallel threads can come from one stage, which simulates real-life automated security control scenarios. Additionally, one of the most advanced testing approaches is the use of the Excessive Data Exposure Fuzz tool, developed to detect excessive data exposure through APIs (Pan *et al.*, 2024). This tool applies a metamorphic testing method to identify cases

where APIs transmit more data than necessary, which can lead to leaks of confidential information.

In turn, AI-enhanced automated bug bounty programmes demonstrate significant potential in detecting vulnerabilities but cannot completely replace ethical hackers. AI systems are able to quickly process large amounts of data, identify common vulnerabilities, and automate certain aspects of security testing. For instance, the Cybersecurity AI framework demonstrates the ability to autonomously solve security problems up to 3,600 times faster than humans, reducing testing costs by an average of 156 times (Mayoral-Vilches *et al.*, 2025). Such a system combines AI agents with human supervision, which provides efficient detection of critical vulnerabilities.

In the context of automation, an important area of improving cyber resilience is the implementation of DevSecOps practices that integrate security at all stages of the software development lifecycle. By automating code review processes, continuous vulnerability monitoring, and configuration control, DevSecOps can quickly detect and correct security issues before product deployment. Static and dynamic analysis tools, dependency checking, automatic container scanning, and integration with CI/CD pipelines help reduce the risk of vulnerability exploitation in a production environment. This approach not only reduces threat response times but also fosters a culture of security responsibility among developers, testers, and DevOps teams.

IDS and IPS systems are also important elements of cyber defence, especially in large corporate networks (Table 2). However, their effectiveness directly depends on the ability to accurately identify threats and minimise the number of false positives. Testing IDS/IPS solutions using modern methods, including AI, can improve the quality of attack detection, adaptability to new threat vectors, and the overall reliability of the security infrastructure.

Table 2. Comparative analysis of IDS and IPS

Characteristic	IDS	IPS
Use	Detection of potentially harmful activity	Detection and automatic block of attacks
Operation principle	Analyses network traffic in passive mode	Actively filters, blocks or redirects traffic
Typical deployment point	Switch mirror port or tap device	In the gap of network traffic (inline)
Reaction to threats	Generates a message or warning	Automatically applies measures (blocking, drop)
Application examples	Monitoring of internal traffic	Protection at the perimeter level
	Security Information and Event Management analytics	Blocking exploits
AI integration	Anomaly detection, correlation with logs	Automatic decision-making, self-learning

Source: compiled by the author based on A. Karambelkar (2025)

Thus, IDS/IPS testing is becoming increasingly important as network attacks grow and architectures become more complex. At the same time, threats are increasingly shifting from the technical level to the user level, where new generation phishing attacks are becoming particularly dangerous. This stage in the evolution of social engineering, dubbed phishing 3.0, involves the use of AI to create personalised messages, deepfake content and fake calls, making it difficult to detect attacks. Modern phishing campaigns are no longer limited to email, as messengers, social media, and video platforms are also actively used. Effective protection involves a combination of technological solutions, raising user awareness, and implementing Zero Trust principles. In this context, special attention is required to protect critical data transmitted through open or cloud environments.

One of the most effective approaches to protecting data in cloud environments is end-to-end encryption. This method ensures data confidentiality by encrypting it on the sender side and decrypting it only on the receiver side, making data secure even if the infrastructure is compromised. However, end-to-end encryption does not protect against all types of attacks. For example, if the encryption keys are not managed properly or if they are leaked, data can become vulnerable. This also affects performance, especially in data-intensive environments. However, when combined with other security mechanisms, such as multi-factor authentication and anomaly detection systems, end-to-end encryption can be an important element of data protection in cloud systems.

As for the vulnerabilities of IoT devices, they remain one of the most significant cybersecurity challenges, as these devices are often weak points in the network infrastructure. Many of them have limited capacity to implement complex security mechanisms such as encryption or multi-factor authentication. Due to the lack of proper protection, they can be used as entry points for attacks, giving attackers access to more critical network elements. One of the key methods of addressing these vulnerabilities is to implement "security by default", which includes automatic firmware updates, strong passwords, data encryption, and restricted access to devices. In addition, it is important to integrate IoT devices into an overall monitoring and anomaly detection strategy that provides quick identification and response

to potential threats. Table 3 shows a comparative analysis of different approaches to cybersecurity testing and protection, where each approach has unique capabilities and limitations.

Table 3. Comparative analysis of approaches to testing and protection in cybersecurity

Approach	Possibilities	Limitations
API testing	Automation of verification	Requires proper configuration and integration with other systems
	Identification of common threats	
	Integration with CI/CD for early detection of vulnerabilities	May not cover new or complex threats
Bug Bounty programmes	Scalability	High dependence on the quality of testers
	Involving a large number of experienced testers	It may not be sufficient for complex or specific threats
	Quick detection of vulnerabilities	Requires constant monitoring
DevSecOps	Integrating security at all stages of development	Requires changes in organisational culture and ongoing resources to support
	Automation of testing processes	
	Continuous monitoring and patching of vulnerabilities before release	It can be difficult to integrate with existing processes
IDS/IPS	Increased ability to detect attacks	May generate false alarms
	Automatic blocking of threats in real time (IPS)	Complexity of setup to minimise errors
	Correlation with other security systems (IDS)	Limited ability to detect new, unknown threats
Phishing	Use of AI to create personalised attacks	Dependence on users (their training and awareness)
	High complexity of detection	Requires constant adaptation of defence mechanisms for new attacks
	Multichannel	
End-to-end encryption	Protects data in transit	Requires proper key management
		May affect productivity
	Ensures data confidentiality even in the case of infrastructure compromise	Does not protect against all types of attacks (from leakage through malware)
IoT security	High relevance for network security	Poor device security
	Integration with other monitoring systems	Limited resources to implement complex security measures
	Support for security principles by default	

Source: compiled by the author based on M. Neelapu & A. Pub (2023), A. Thool & C. Brown (2025)

Comparing AI and pentesting to other modern approaches to testing and protection, such as API testing, Bug Bounty, DevSecOps, IDS/IPS, end-to-end encryption, IoT security, and phishing, it is possible to note that each has a specific role in cyber defence. Unlike most automated or process-oriented solutions, AI tools provide threat detection through adaptability and scalability, while pentesting remains indispensable for in-depth analysis of complex, context-sensitive vulnerabilities. As a result, AI and pentesting do not compete with other methods, but rather fill in the gaps left by those methods or need to be supplemented in the face of real-world threats, making them important components of a hybrid security approach.

DISCUSSION

The study determined that the use of AI in automated vulnerability testing ensures high efficiency in detecting typical threats at early stages, especially in time-limited conditions. Similarly, the study by A. Pispa & K. Halunen (2024) proved the importance of implementing structured approaches, including the Open Worldwide Application Security Project, to classify and assess vulnerabilities of AI systems throughout their life cycle. Both papers recognise the critical role of AI in security, but the present focus is shifted to practical applications, while the aforementioned paper focuses on methodological support.

The results showed that traditional pentesting remains critical for detecting complex and context-dependent vulnerabilities that are not available to fully automated AI solutions. At the same time, P. Wylie & K. Crawley (2020) reviewed key pentesting tools (Kali Linux, Metasploit, Wireshark) and the importance of hands-on labs for practising skills. Both papers agree on the importance of pentesting, but the current one focuses on analysing the strategic benefits, while the other one points out the technical implementation of individual tools.

The present study determined that automated API testing using AI can effectively detect typical threats related to authentication, authorisation, and data leaks, and it is also integrated into CI/CD for early detection of vulnerabilities. In contrast, Z. Wang *et al.* (2024) analysed the detection of logical errors in REST (Representational State Transfer) APIs that remain outside the scope of traditional testing methods by building a logic model and feedback-based analysis. Both approaches agree on the importance of automation and deep API analysis, but the present study addressed typical automation and CI/CD integration, while the other focuses on modelling API logic and finding complex logic defects.

The study analysed the use of various ML algorithms for vulnerability detection, in particular RF, k-NN and NBC, incorporating their capabilities, limitations and feasibility in cybersecurity. In the work of M. Andani *et al.* (2025), the effectiveness of these algorithms was also investigated based on various attributes. Both works emphasised the high performance of RF and the general suitability of ML algorithms for classification tasks. However, the current study focused on their application in the field of cybersecurity, where not only accuracy but also interpretability, speed, and attack resistance are critical, while the present study uses ML in a less dynamic environment.

Similarly, to a study by R. Martin *et al.* (2024), the present research addressed the use of DT, SVM and NN algorithms for cyber threat detection. Both studies confirmed the effectiveness of these algorithms, in particular the accuracy of SVM and the ability of NN to detect complex patterns. At the same time, the current study emphasised the interpretability, performance, and robustness of models in the field of vulnerability detection, while the aforementioned study focused on their application in antivirus systems without a deep analysis of limitations.

Moreover, the results showed that AI-enhanced bug bounty programmes effectively automate the detection of common vulnerabilities and reduce costs, but do not replace ethical hackers in complex cases. J. Silomon *et al.* (2022) also emphasised the role of bug bounty programmes, namely crowdsourced platforms, as tools for cooperation and regulation in cyberspace. Both approaches acknowledge the effectiveness of bug bounties, but the work conducted focuses on technical automation, while the other on the socio-political aspect.

Compared to the study by S. Riaz *et al.* (2025), where DevSecOps was considered mainly as a concept of integrating security into the entire software lifecycle with a review of current literature, AI tools and trends (ML, zero-trust, cloud-native security), this study focused on the implementation of DevSecOps in the CI/CD environment. It considered aspects of automating code analysis, dependency checking, container scanning, and continuous vulnerability monitoring until release. Thus, these works correlate with the importance of continuous security testing and integration of controls into development, but the study complements the existing review by analysing the use of AI in the DevSecOps context, emphasising the effectiveness of automated approaches at the stages of vulnerability detection and API security.

S. Arifin *et al.* (2025) considered a hybrid encryption system that combines a unimodular cypher and the Advanced Encryption Standard in an electronic codebook mode, with an emphasis on cryptographic performance and statistical characteristics such as encryption time and entropy. In comparison, the current study focused on end-to-end encryption as part of a data protection strategy in cloud environments. Both approaches share the common goal of ensuring data confidentiality, but the current study additionally focuses on key management and integration with other security mechanisms, making it more applicable to modern cloud infrastructures.

While the study by S. Bharti (2024) addressed the integration of IDS/IPS into the operating system environment as a means of increasing their resilience to cyber threats, the present research analysed these systems in the context of corporate networks and their testing with the use of AI. While both approaches recognise the critical role of IDS/IPS in modern cybersecurity, this study focuses on improving them through AI, particularly in terms of anomaly detection, adaptability, and reducing false positives, which is especially relevant in the context of complex network infrastructure.

In this study, DL is considered an effective tool for automated analysis of unstructured data, such as event logs or network traffic, with a special focus on detecting multi-vector attacks using recurrent neural networks. In turn, the study by A. Almajali *et al.* (2024) also used DL, particularly the Deep Reinforcement Learning approach, to create an agent capable of automated exploitation of a specific vulnerability. Thus, both works confirmed the effectiveness of DL in vulnerability detection and exploitation, but the present study focused on the active component of the attack, the formation of reward-based exploit actions, while the work conducted on event and data analysis for the purpose of classifying and early detection of threats.

Current results have shown that AI significantly improves the efficiency of automated testing, especially in CI/CD environments, providing accuracy, speed, and detection of common vulnerabilities. A. Awad *et al.* (2024) also highlighted the benefits of AI to improve the quality of automated testing and overcome the limitations of traditional methods. Both studies agree on the potential of AI in automation, but the current study goes deeper into the integration into dynamic development processes.

The conclusions of the present study and those of P. Singh *et al.* (2025) showed different approaches to vulnerability detection, in particular in the context of phishing attacks. While the work conducted in this study focuses on the use of AI to create personalised phishing attacks that use deepfake content and fake calls, which makes them difficult to detect, the aforementioned study addressed automation of intelligence through the Recon Automator tool, which helps to quickly find vulnerabilities in infrastructure, through data collection and analysis. However, their work does not pay as much attention to social engineering, which is a major factor in modern phishing attacks. At the same time, the current study emphasised the complexity of detecting phishing 3.0, the need to constantly adapt defence mechanisms and integrate Zero Trust principles to improve overall system security.

Additionally, the study by B.-H. Liang *et al.* (2025) discussed the main vulnerabilities of IoT devices, in particular, vulnerability to Denial-of-Service attacks and malware detection methods. It was found that some devices are immune to certain viruses. The

current study also highlighted the limited ability of IoT devices to implement sophisticated security measures, such as encryption and multi-factor authentication, as well as their vulnerability due to the lack of regular updates. At the same time, it highlighted the importance of regular updates and monitoring to detect anomalies in networks with IoT devices.

Thus, this study has shown that the use of AI in automated vulnerability testing significantly increases the efficiency of detecting common threats, especially in time-sensitive environments. At the same time, traditional pentesting remains important for detecting complex and context-dependent vulnerabilities. The use of AI for automated API testing, integration into CI/CD environments, and the application of ML algorithms demonstrated high accuracy and speed in detecting vulnerabilities. AI proved particularly effective in detecting typical cybersecurity threats such as authentication, authorisation, and data leaks.

CONCLUSIONS

The analysis of the effectiveness of modern AI-based approaches to vulnerability detection has shown that the use of ML algorithms significantly improves the process of threat detection compared to traditional methods. Algorithms such as DT, SVM, NBC, NN, DL, k-NN, and RF have different capabilities and limitations, which can be used for further combination to efficiently solve vulnerability detection tasks. While some algorithms demonstrated high accuracy, others were distinguished by their speed or ability to process big data. At the same time, none of the algorithms in question is universal, and their choice depends on the specifics of the task and available resources. For instance, lighter models, such as DT or NBC, are suitable for standard tasks, while more complex models, such as DL, are substantial in the detection of non-trivial attacks but require significant computing power. At the same time, the use of hybrid systems that combine several approaches provided better results in the face of complex cyber threats.

Despite the growing role of AI in cybersecurity, traditional pentesting methods remain important due to their ability to address complex contexts and atypical situations. Pentesting, which involves simulating real attacks, is important for identifying complex vulnerabilities that automated AI systems are not always able to detect. However, due to its speed and ability to adapt, AI is becoming an effective complement to pentesting, automating testing and helping to identify common threats. The hybrid ML model, which combines RF and SVM, achieved high overall accuracy (93%) and perfect prediction accuracy (100%), which indicates its ability to avoid false alarms, but has limitations in completeness (83%), missing 17% of vulnerabilities, which is inferior to pentesting (92%). The model's advantage is speed and automation, but its limitations are the dependence on the quality of synthetic data and insufficient sensitivity to context, which confirms the need for a hybrid approach for comprehensive protection.

AI tools, such as those for API testing or vulnerability detection through bug bounty programmes, continue to improve security, but cannot completely replace the human factor. The DevSecOps approach and integration of AI into the development process can reduce risks through automation, while the use of technologies such as end-to-end encryption increases data protection in cloud environments. Additionally, technologies such as IDS/IPS play an important role in detecting and blocking attacks in real time but can have problems with false positives and limited ability to detect new threats. Security for the increasingly prevalent IoT devices requires integration with other security systems to protect against device-specific vulnerabilities, and they often have poor security and limited upgrade capabilities.

The limitations of the study include the comparison of different vulnerability testing approaches being based on general principles and theoretical data, which may limit the accuracy of the conclusions in real-world environments where the specifics of each system may matter. In addition, due to the rapid development of technology, new tools and methods can quickly become outdated, and the results may not be relevant for some types of testing. Further research should focus on integrating AI into the broader context of cybersecurity, testing new technologies such as 5G, and developing adaptive systems that can consider the specific requirements of each organisation.

ACKNOWLEDGEMENTS

None.

FUNDING

None.

CONFLICT OF INTEREST

None.

REFERENCES

[1] Akinyemi, A.M., & Sims, S. (2025). Role of artificial intelligence in modern cybersecurity vulnerability management practices. *World Journal of Advanced Research and Reviews*, 26(1), 555-584. doi: 10.30574/wjarr.2025.26.1.1028.

[2] Almajali, A., Al-Abed, L., Yousef, K.M., Mohd, B.J., Samamah, Z., & Abu Shhadeh, A.I. (2024). Automated vulnerability exploitation using deep reinforcement learning. *Applied Sciences*, 14(20), article number 9331. doi: 10.3390/app14209331.

[3] Almheiri, H.M., Ahmad, S.Z., Abu Bakar, A.R., & Khalid, K. (2024). Artificial intelligence capabilities, dynamic capabilities and organizational creativity: Contributing factors to the United Arab Emirates government's organizational performance. *Journal of Modelling in Management*, 19(3), 953-979. doi: 10.1108/JM2-11-2022-0272.

[4] Andani, M., Triloka, J., Irianto, S.Y., & Nugroho, H.W. (2025). Comparison of K-nearest neighbor, naive bayes, random forest algorithms for obesity prediction. *SinkrOn*, 9(1), 502-510. doi: 10.33395/sinkron.v9i1.14478.
[5] Arifin, S., Wijonarko, D., Faisal, M., Pratama, M.N., & Prasetyo, P.W. (2025). Text data security through double encryption: Implementation of unimodular hill cipher and advanced encryption standard. *International Journal on Advanced Science Engineering and Information Technology*, 15(2), 444-455. doi: 10.18517/ijaseit.15.2.20424.
[6] Awad, A., Qutqut, M.H., Ahmed, A., Alhaj, F., & Almasalha, F. (2024). Artificial intelligence role in software automation testing. In *Conference: 2024 international conference on decision aid sciences and applications (DASA)* (pp. 1-6). Manama: IEEE. doi: 10.1109/DASA63652.2024.10836630.
[7] Bharti, S. (2024). Intrusion detection and prevention systems (IDS/IPS) for OS protection. *Interantional Journal of Scientific Research in Engineering and Management*, 8(4), 1-5. doi: 10.55041/IJSREM31718.
[8] Capelli, G., Verdi, D., Frigerio, I., Rashidian, N., Ficorilli, A., Grasso, S.V., Majidi, D., Gumbs, A.A., Spolverato, G., & Taher, H. (2023). White paper: Ethics and trustworthiness of artificial intelligence in clinical surgery. *Intelligence & Robotics*, 3(2), 111-122. doi: 10.20517/ais.2023.04.
[9] Cruz, E. (2025). Sustaining human vulnerability at the crossroads of the sciences of the self, artificial, and spiritual intelligence. *Christian Perspectives on Science and Technology*, 3. doi: 10.58913/OWHV7073.
[10] Karambelkar, A. (2025). Next generation firewall using IPS & IDS. *International Journal for Research in Applied Science and Engineering Technology*, 13(4), 2868-2874. doi: 10.22214/ijraset.2025.68804.
[11] Kudriashov, A. (2024). Artificial intelligence and security in 5G and 6G mobile technologies. *Computer-Integrated Technologies: Education, Science, Production*, 54, 236-242. doi: 10.36910/6775-2524-0560-2024-54-29.
[12] Kumar, A. (2025). Ethical hacking and penetration testing. *International Scientific Journal of Engineering and Management*, 4(4). doi: 10.55041/ISJEM02790.
[13] Liang, B.-H., Hwang, R.-H., Lin, J.-Y., & Chen, H.-H. (2025). Comprehensive vulnerability detection and malware infection testing strategies for IoT devices. *IEEE Internet of Things Journal*, 12(12), 20556-20571. doi: 10.1109/JIOT.2025.3543819.
[14] Martin, R., Pava, R., & Mishra, S. (2024). Analyzing machine learning algorithms for antivirus applications: A study on decision trees, support vector machines, and neural networks. *Issues in Information Systems*, 25(4), 455-465. doi: 10.48009/4_iis_2024_135.
[15] Mayoral-Vilches, V. *et al.* (2025). CAI: An open, bug bounty-ready cybersecurity AI. *Arxiv.* doi: 10.48550/arXiv.2504.06017.
[16] Nasir, M., & Pomeroy, J. (2025). *Ethical hacking meets AI: Revolutionizing vulnerability assessments and penetration testing*. doi: 10.13140/RG.2.2.25822.55368.
[17] Neelapu, M., & Pub, A. (2023). Enhancement of software reliability using automatic API testing model. *International Journal of Multidisciplinary Research and Growth Evaluation*, 4(3), 1113-1117.
[18] Nguyen, H.P., Zhi, C., Hasegawa, K., Fukushima, K., & Beuran, R. (2024). PenGym: Pentesting training framework for reinforcement learning agents. In *Proceedings of the 10th international conference on information systems security and privacy* (pp. 498-509). Rome: ScitePress. doi: 10.5220/0012367300003648.
[19] Pan, L., Cohney, S., Murray, T., & Pham, V.-T. (2024). EDEFuzz: A web API Fuzzer for excessive data exposures. In A. Paiva & R. Abreu (Eds.), *Proceedings of the 46th IEEE/ACM international conference on software engineering* (article number 45). New York: Association for Computing Machinery. doi: 10.1145/3597503.3608133.
[20] Pispa, A., & Halunen, K. (2024). Comprehensive artificial intelligence vulnerability taxonomy. *European Conference on Cyber Warfare and Security*, 23(1), 379-387. doi: 10.34190/eccws.23.1.2157.
[21] Pratama, D., Suryanto, N., Adiputra, A.A., Le, T.-T., Kadiptya, A.Y., Iqbal, M., & Kim, H. (2024). CIPHER: Cybersecurity intelligent penetration-testing helper for ethical researcher. *Sensors*, 24(21), article number 6878. doi: 10.3390/s24216878.
[22] Putri, A.I., Husna, N.A., Mella, N., Arba, M.A., Aisyi, N.R., Pramesthi, C.H., & Irdayusman, A.S. (2024). Implementation of K-nearest neighbors, naïve bayes classifier, support vector machine and decision tree algorithms for obesity risk prediction. *Public Research Journal of Engineering Data Technology and Computer Science*, 2(1), 26-33. doi: 10.57152/predatecs.v2i1.1110.
[23] Riaz, S., Asif, A., Khan, Y., Ibrar, M., Afzal, S., Hamid, K., Gul, S., & Iqbal, M.W. (2025). Software development empowered and secured by integrating a DevSecOps design. *Journal of Computing & Biomedical Informatics*, 8(2).
[24] Shvets, F., Soroka, V., Zoshchuk, V., Shvets, M., & Moroz, O. (2024). Artificial intelligence technologies in education: Opportunities and prospects for use. *Bulletin National University of Water and Environmental Engineering*, 2(106), 271-281. doi: 10.31713/ve2202425.
[25] Silomon, J., Hansel, M., & Schwartz, F. (2022). Bug bounties: Between new regulations and geopolitical dynamics. *International Conference on Cyber Warfare and Security*, 17(1), 298-305. doi: 10.34190/iccws.17.1.21.
[26] Singh, P., Agrawal, P.P., & Dolai, S. (2025). Recon automator: Enhancing cybersecurity reconnaissance with automation. *International Journal for Research in Applied Science and Engineering Technology*, 13(3), 2729-2735. doi: 10.22214/ijraset.2025.67932.

[27] Teo, S.A. (2025). Artificial intelligence, human vulnerability and multi-level resilience. *Computer Law & Security Review*, 57, article number 106134. doi: 10.1016/j.clsr.2025.106134.
[28] Thool, A., & Brown, C. (2025). Integrating DAST in Kanban and CI/CD: A real world security case study. *ArXiv*. doi: 10.48550/arXiv.2503.21947.
[29] Trofymenko, O., Sokolov, A., Chykunov, P., Akhmametieva, H., & Manakov, S. (2024). AI in the military cyber domain. *Technologies and Engineering*, 4, 85-92. doi: 10.30857/2786-5371.2024.4.8.
[30] Wang, N., Pan, S., & Wang, Y. (2025). How can artificial intelligence capabilities empower sustainable business model innovation? A dynamic capability perspective. *Business Process Management Journal*. doi: 10.1108/BPMJ-11-2024-1045.
[31] Wang, Z., Tian, W., & Cui, B. (2024). RESTlogic: Detecting logic vulnerabilities in cloud REST APIs. *Computers, Materials & Continua*, 78(2), 1797-1820. doi: 10.32604/cmc.2023.047051.
[32] Wylie, P., & Crawley, K. (2020). Building a pentesting lab. In P. Wylie & K. Crawley (Eds.), *The pentester blueprint* (pp. 65-81). London: John Wiley & Sons. doi: 10.1002/9781119684367.ch5.

Можливості та обмеження штучного інтелекту в тестуванні вразливостей: ефективність підходу на противагу традиційному пентестингу

Ігор Руденко

Магістр
Національний юридичний університет імені Ярослава Мудрого
61024, вул. Григорія Сковороди, 77, м. Харків, Україна
https://orcid.org/0009-0008-3582-3951

Анотація. Актуальність роботи зумовлена зростаючими загрозами в кібербезпеці, що вимагають удосконалення методів виявлення вразливостей у програмному забезпеченні, зокрема за допомогою новітніх технологій, таких як штучний інтелект (ШІ). Мета дослідження – визначити ефективність застосування ШІ для тестування вразливостей як альтернативи традиційним методам тестування безпеки, зокрема пентестингу. Для досягнення мети було проведено аналіз алгоритмів машинного навчання, розроблено гібридну модель для виявлення вразливостей, а також проаналізовано застосування систем виявлення та запобігання вторгненням, автоматизованих підходів до тестування програмного забезпечення та захисту прикладних інтерфейсів. Дослідження виявило, що дерева рішень швидко класифікують трафік, але перенавчаються; метод опорних векторів точно аналізує журнали, але чутливий до налаштувань; наївні баєсівські класифікатори ефективно фільтрують повідомлення, але обмежені припущеннями; нейронні мережі та глибинне навчання виявляють складні загрози, але потребують багато даних; алгоритм k найближчих сусідів підходить для малих систем, але повільний; випадкові ліси точні в аналізі коду, але менш інтерпретовані. ШІ швидкий і масштабований, але обмежений у розумінні контексту та реагуванні на нові загрози. Результати показали, що пентестинг переважає у виявленні складних вразливостей, а гібридна модель машинного навчання досягла точності 93 % та точності передбачень 100 %, але пропустила 17 % вразливостей. Зі свого боку, технології кібербезпеки, як тестування прикладних програмних інтерфейсів, програми винагороди за помилки, інтеграція безпеки в розробку, системи виявлення вторгнень, наскрізне шифрування, захист інтернету речей і протидія фішингу, також ефективні, але потребують адаптації до нових загроз. Результати дослідження можуть бути використані компаніями, що займаються кібербезпекою, розробниками програмного забезпечення та інженерами з безпеки для вдосконалення процесів тестування вразливостей і захисту від кіберзагроз, зокрема за допомогою інструментів на базі ШІ

Ключові слова: алгоритми машинного навчання; гібридні моделі аналізу; системи виявлення та запобігання вторгнень; автоматизація перевірки програмного забезпечення; захист прикладних інтерфейсів; сценарії імітації атак