From Threat Hunting to Prevention: The Role of Python in Cybersecurity Defense

A. Vinitha[1], Kiran Kumari Patil[2], K. Reena[3], D. Yobu[4], D. Vikram[5] and S. P. Santhoshkumar[6*]

{viniarch26@gmail.com[1], ranjaniramesh1120@gmail.com[2] , reenaanbhazhagan@gmail.com[3], yobud2017@gmail.com[4], dvikram91@gmail.com[5], spsanthoshkumar16@gmail.com[6] }

Assistant Professor, Department of Computer Science, Karpagam Academy of Higher Education, Coimbatore, Tamil Nadu, India.[1]

Professor, Department of Computer Science and Engineering, CMR university, Bangalore, Karnataka, India[2]

Assistant Professor, Department of Computer Science and Business Systems, Knowledge Institute of Technology, Kakapalyam, Salem, Tamil Nadu, India[3]

Assistant Professor, Department of Computer Science and Engineering (Cyber Security), J.J College of Engineering and Technology, Trichy, Tamil Nadu, India[4]

Assistant Professor, Department of Artificial Intelligence and Data Science, Rathinam Technical Campus, Coimbatore, Tamil Nadu, India[5]

Assistant Professor (SG), Department of Computer Science and Engineering, Vel Tech Rangarajan Dr. Sagunthala R&D Institute of Science and Technology, Avadi, Chennai, Tamil Nadu, India[6]

Abstract. Python is increasingly being used to check and mitigate threats in the fast-moving niche of cyber security. This article explores the use of Python to identify and support remediation of common cyberthreats such as malware, phishing, unauthorized access via open ports, password use weakness, and network anomalies. We use these algorithms and consider execution time, detection accuracy, code robustness and possible vulnerabilities, to evaluate the ability of a Python-based algorithm to detect these threats. Despite the speed that Python is developed and deployed, we found through our research that threat detection efficacy varies by the attack vector. This article summarises a comprehensive comparison of the detection implementations, and the benefits and limitations are discussed to emphasise the necessity of adaptive changing in security in cyber warfare.

Keywords: Cybersecurity, Denial-of-Service, Threat Detection, Packet Analysis, Network Analysis, Hashing.

1 Introduction

With more productivity, connectivity, and automation in every industry, the rapid digitization of industries has entirely revolutionized the way businesses work. But this change has also led to a like never before rise in cyberattacks, and cybersecurity has become one of the most pressing issues of the digital age [1]. Advanced attacks aimed at flaws in systems, networks and applications are becoming increasingly hazardous for governments, corporations, and people. The critical need for effective security mechanisms is emphasised by the fact that the exploitation of such weaknesses has often led to data breaches, identity theft, financial loss, damage to reputation and widespread service interruptions [2].

ICITSM-Part II 2025, April 28-29, Tiruchengode, India
Copyright © 2025 EAI
DOI 10.4108/eai.28-4-2025.2358054

Traditional malevolent actions are no longer the only cyberthreats; their scope, variety and intelligence has grown [3]. These days, more advanced techniques are used by attackors including automated port scanning, social-engineering phishing campaigns, polymorphic malware, and exploiting weak authentication procedures [4]. In addition, as the cloud economy expands and the number of connected devices becomes more widespread, the attack surface has been widened, making it easier for the enemy to breach the old defenses [5]. To get ahead of the attackers, the cybersecurity experts have increasingly relied on automation, scripting and intelligent detection methods in response to these difficulties [6].

Python has become a key part of modern cybersecurity defense within the many cybersecurity tools available [6]. Even complex tasks such as log analysis, vulnerability scanning, malware analysis and intrusion detection can be performed effectively with the help of its readability, simplicity and adaptability [7]. Python is especially suitable for threat hunting and preventive defense tactics because of its powerful library and framework availability and great community support [8]. Security professionals can boost their accuracy and adaptability in countering cyber threats with rapid prototyping of security solutions, automation of the tedious process, establishing cutting-edge machine learning algorithms in neural network systems for detection [9], [10].

In order to bridge the gap between manual threat hunting and automated prevention this paper accentuates the significance of Python, not as a supporting tool, but as a central part of enhancing modern-day cybersecurity defense mechanisms. This paper examines the transformative nature of Python in dealing with five major types of cyberattacks, which includes malware infection, phishing attempts, unauthorized access using open ports, weak password attacks, and network anomalies. For every category, to investigate various methods of python-based detection systems, to compare different methods, to evaluate different methods for effectiveness of detection and to analyse the computational performance of different methods.

1.1 Expanding Attack Surfaces in the Digital Era

The rise of cloud computing, IoT and remote working has created a bigger digital attack surface opening more people and organizations to increasingly frequent and serious cyber threats such as ransomware, phishing, DDoS and data breaches, which can cause significant financial and reputational damage and make it necessary to have faster and smarter ways to defend against them [11]. Complexity: Python's powerful ecosystem and ease of use make it possible to develop adaptive, automated, and artificial intelligence (AI)-driven threat detection systems, making it an essential tool for both proactive threat hunting and preventative cybersecurity defense [12].

1.2 Why Use Python in Cybersecurity?

Who doesn't love Python, its versatleness, ease-of-use and powerful toolkit have made it the programming language of choice in modern cybersecurity [13]. Python offers a balance simplicity and performance which makes it ideal for both quick prototyping and deployed security solutions, as opposed to low-level languages that take a huge amount of development time. It is an essential tool for the defense against cybersecurity due to its multi-dimensional role of threat identification, digital forensics, malware analysis, penetration testing, and automatic defence mechanisms [14].

Readability and Simplicity: Security researchers can quickly prototype and test detection techniques thanks to Python's high-level syntax and straightforward design that isn't or need not be held back by complex programming constructs. This helps accelerate development cycles and ensures that changes to new cyberthreats can be made in a timely manner.

Wide-ranging Library: Python has extensive support with libraries, which is one of the best features of Python. Advanced capabilities, such as network analysis, packet inspection, cryptographic hashing, behavioral monitoring, and even threat detection with the help of machine learning, are all made possible by special packages. By removing the development overhead these libraries allow researchers to focus on solving security problems instead of building tools from scratch.

Capabilities for Automation: Cyber Security: Python is for cyber security automation to minimize tedious and time-consuming tasks. From malware reverse engineering to vulnerability scanning to log parsing to penetration testing, Python scripts make everything easier for analysts, who can focus on higher-level threat intelligence and strategic defense planning.

Integration of AI and Machine: Detection systems are needed to be smarter thanks to the increasing sophistication of cyberattacks. The reason why we can build anomaly detection models, predictive threat analysis and adaptive defense systems is because Python integrates so well with artificial intelligence frameworks. Python is poised to become a key enabler in the creation of next-generation, AI-powered cyber-security solutions thanks to this ability.

Compatibility Across Platforms: Security experts are able to develop scalable and portable solutions due to Python's compatibility with different operating systems. Python ensures uniformity and adaptability in diverse infrastructures be it Windows, Linux or Mac OS environments. This is necessary for the security management at the enterprise level [15].

By combining these characteristics Python has evolved into more than just another programming language, it is a fundamental technology in the field of cybersecurity and is bridging the gap between automated defense and manual threat analysis.

2 Cyber Threat Environment

The world of cybersecurity threats is changing quickly, and attackers are employing increasingly complex methods to get around established protection measures.

Malicious software, or malware, is intended to infect and harm a system. Ransomware, trojans, worms, viruses, and spyware are among them. Attackers spread malware using infected USB devices, rogue websites, email attachments, and software flaws.

Impacts include:

- Encrypting files and requesting a ransom (ransomware attacks);
- Stealing private information, such as banking credentials;
- Cryptojacking via stealing system resources

Conventional detection techniques include behavioral-based detection, which looks for odd activity, and signature-based detection, which compares malware signatures in a database.

Phishing Attacks: Phishing is a type of social engineering attack in which hackers pose as trustworthy organizations in order to fool consumers into disclosing private information. Usually, spoof websites, phony login pages, and bogus emails are used in these assaults.

Impact:

- Identity theft (password, username, and personal information theft)
- Financial fraud (gaining access to credit card information and bank accounts)
- Corporate espionage, which involves obtaining private company information without authorization.

Methods of detection:

- Analysis of domain and URL reputation
- Classification using machine learning; analysis of email headers and metadata
- (Unauthorized Access) Port Scanning: To find open ports and weak services operating on a system, attackers employ port scanning. Once discovered, an open port might be used to launch more attacks, gain unauthorized access, and compromise data.

Effects include:

- Getting access to private data (such as unsecured databases)
- Starting assaults using remote code execution (RCE)
- Making use of system flaws, such as out-of-date software

Methods of detection:

- Tools for network traffic monitoring; intrusion detection systems (IDS); and programs for Python-based port scanning detection
- Password Exploitation with Weakness: Because so many users continue to use weak or frequently used passwords, they are prime targets for brute-force attacks. Automated scripts are used by cybercriminals to guess passwords and access user accounts.

Impact:

- Unauthorized transactions and financial fraud;
- Credential stuffing (using credentials that have been compromised on several accounts);
- User account compromise

Prevention strategies include:

- Implementing strict password regulations
- Using tools for detecting password strength based on Python; putting two-factor authentication (2FA) into practice;
- Identifying Network Anomalies: Network irregularities may be a sign of insider threats, DDoS attacks, or data espionage. Attackers frequently conceal their actions within typical network traffic, making identification difficult.

Impact:

- Disrupting business operations through service downtime
- Unauthorized access to sensitive corporate data
- Financial losses due to fraudulent activities

Detection methods include:

- AI-driven behavioral analysis;
- Real-time network monitoring; and
- Python-based anomaly detection programs.

The Need for Automatic Threat Identification In the face of contemporary cyberthreats, traditional cybersecurity techniques are losing their effectiveness. With the advent of sophisticated phishing kits, automated attack scripts, and malware powered by artificial intelligence, manual security monitoring is no longer adequate. Automated threat detection systems are required to do this.

Reduce Response Time: Early detection of cyberattacks allows for quicker mitigation.

Boost Accuracy: AI-capable Python scripts are able to spot hidden patterns in network traffic.

Handle Large Data Volumes: Real-time threat detection is made possible by machine learning models that have been trained on large datasets. By providing automated scanning, anomaly detection, and real-time monitoring, Python-based security tools strengthen an organization's resilience against online attacks.

Attacks and Detection Mechanisms: Cyber threats have evolved significantly, targeting vulnerabilities in networks, applications, and users. This paper focuses on five common attack types: malware infection, phishing, unauthorized access via open ports, weak passwords, and network anomalies. Each attack has a corresponding detection mechanism, implemented in Python.

Viruses, trojans, ransomware, and spyware are examples of malicious software that is intended to interfere with, harm, or obtain unauthorized access to a system. This is known as malware detection. Malware frequently hides in scripts, macros, or executable files.

Detection Mechanism: Hash-based malware detection is a popular method that compares a file's hash (MD5, SHA-256) with a database of known malware hashes. The file is marked as malicious if the hash matches. Sandboxing, machine learning-based anomaly detection, and behavioral analysis are more sophisticated methods.

Phishing URL Detection: Phishing attacks utilize phony emails or websites to deceive users into divulging private information, such as banking information and passwords. Social engineering techniques and phony domains that mimic authentic websites are frequently used in these attacks.

Detection Mechanism I: Python programs can be used to determine whether a password is one of the weak passwords that are frequently used, such as 123456, password, admin, or qwerty. More complex systems enforce password strength guidelines that call for a combination of special characters, numerals, capital letters, and lowercase letters.

Detection Mechanism II: A simple technique for identifying phishing attempts is to search for questionable terms such as free, win, bank, login, and verify on websites. More sophisticated techniques employ machine learning algorithms that examine the age of the domain, HTML content, and URL structure. AI-powered detection is crucial since phishing techniques are always changing.

Unauthorized Access (Port Scanning): Attackers employ port scanning as a reconnaissance tool to find open ports on a target system. Services (such FTP, SSH, and MySQL) that are exposed by open ports may be used by hackers to gain unwanted access.

Method of Detection: Scanning a system's frequently used ports (22, 80, 443, 3306, etc.) to see if they are open is a straightforward detection technique. More sophisticated methods log and stop questionable scanning efforts using network firewalls and intrusion detection systems (IDS).

Exploitation of Weak Passwords: One of the most significant security flaws is still weak passwords. Brute-force and dictionary attacks are used by attackers to guess user passwords and access accounts.

Network Anomaly Detection: A possible cyberthreat is indicated by network anomalies including Distributed Denial-of-Service (DDoS) assaults, odd login attempts, and data exfiltration.

Detection Mechanism: A straightforward Python-based technique looks for abnormally high or low activity by tracking network traffic patterns. By identifying anomalous traffic flow, machine learning models can be trained to stop attacks before they do any harm.

3 Attack Comparison

The many forms of cyberattacks differ in their complexity, ease of execution, and detectability. A comparison is provided in the following table 1

Table 1. Comparison Table for various attacks.

Attack Type	**Ease of Execution**	**Detection Complexity**	**Impact Level**	**Common Prevention Methods**
Malware	Medium	High	Critical	Antivirus, Behavioral Analysis
Phishing	High	Medium	High	AI-Based URL Analysis
Port Scanning	High	Low	Medium	Firewalls, IDS
Weak Passwords	High	Low	High	Strong Password Policies
Network Anomalies	Medium	High	Critical	AI-Based Traffic Analysis

3.1 Algorithms for Attack Detection

Python is a powerful language for cybersecurity since it has features for file analysis, network monitoring, and anomaly detection. The following is a list of the detection methods applied to each attack:

Hashing-Based Malware Detection Algorithm

Using open (file_path, "rb") as the file, importhashlib defis_malicious(file_path, known_hashes):

The hashlib.md5 file_hash is(file.read()).

Return file_hash in known_hashes using hexdigest()

This technique compares the hash of a file to hashes of known viruses.

Algorithm for Detecting Phishing URLs

Return any (keyword in url.lower() for keyword in suspicious_keywords) in defis_phishing(url, suspicious_keywords).

checks for questionable terms like "win" or "bank" in a URL.

3.2 Port Scanning Algorithm

import socket

open_ports = [] in defscan_ports(target_ip, ports)

withsocket for port in ports.socket(socket.SOCK_STREAM, socket.AF_INET) as sock: sock.settimeout (0.5)

Open_ports.append(port) ifsock.connect_ex((target_ip, port)) == 0

returnopen_ports

determines whether a list of ports is open by scanning it.

Inadequate Password Recognition Password in common_passwords is returned by the algorithm defis_weak_password(password, common_passwords).

determines whether a password is on a list of weak passwords.

Identifying Network Anomalies Data for data in traffic_data if data > threshold is returned by the algorithm defdetect_anomalies(traffic_data, threshold).

detects anomalous network traffic surges.

4 Result Analysis

The paper has shown that Python-based detectors are highly performant and competitive for certain attack classes. All the malware and weak password detection algorithms were found to be highly accurate and stable, with limited vulnerabilities. Unfortunately, the phishing URL detection system, which was based on simple keyword matching, was not as accurate, and easily tricked. Probe succeeded to detecting the ports OPEN, despite firewalls or filtering system through the network biases. The detection of network anomalies is beneficial to some but for improving the rate of accuracy and minimization of false positives, a more refined analysis than only detection on the hot spot alone is necessary. In general, the detection of threats using detection algorithms may not be an entirely effective solution mainly due to the dynamic nature and evolution of threats in cyber-security although Python's framework is found versatile and feasible for developing such systems.

5 Conclusions

In this paper, we were studied the Python - based strategies in order to removal some of these network threats such as malwares, spyware, phishing, open port sniffer, weak password realization which are all caused by problems in firewalls and network system. We showed with our research that Python’s rich library support, automation features, and machine learning integration, made it suitable for cyber security. The analysis of the detection mechanisms suggests that hash-based malware detection and weak password are very effective, and can achieve both accurate and prompt detections. But in the case of phishing URL detection, since the detection process is based on keyword in URL matching, there were limitations in dealing with advanced types of phishing attacks. The port scanning detection worked very well despite requiring network and firewall configuration. Network anomaly detection shows potential, but more sophisticated algorithms are needed to boost accuracy and minimize false positives.

The performance of Python as a platform for cybersecurity touchpoints on the need for improvements and updating to patterns of cyber-attacks and threats even as it provides robust and highly adaptable realtime threats detection frameworks. In the future, further research needs to consider the AI based-methodologies, the behavior analysis and real-time adaptive security measures in order to increase the accuracy level of detection and the system robustness. Automated threat detection and proactive cybersecurity policies should be implemented together by organizations to provide comprehensive protection against increasingly sophisticated cyber security threats.

References

[1] Adib Bin Rashid, M.D., & Karim Kausik, M.D.A. (2024). AI revolutionizing industries worldwide: A comprehensive overview of its diverse applications. Hybrid Advances, 7, 100277. https://doi.org/10.1016/j.hybadv.2024.100277

[2] Saeed, S., Altamimi, S.A., Alkayyal, N.A., Alshehri, E., & Alabbad, D.A. (2023). Digital transformation and cybersecurity challenges for businesses resilience: Issues and recommendations. Sensors, 23(15), 6666. https://doi.org/10.3390/s23156666

[3] Breached Company. (2025). DOGE software engineer's computer infected by info-stealing malware: A deep dive into the incident and its implications. Retrieved from https://breached.company/doge-software-engineers-computer-infected-by-info-stealing-malware-a-deep-dive-into-the-incident-and-its-implications/

[4] Varadharajan, V., & Suri, N. (2024). Security challenges when space merges with cyberspace. Space Policy, 67, 101600. https://doi.org/10.1016/j.spacepol.2023.101600

[5] Almutairi, M., & Sheldon, F.T. (2025). IoT–cloud integration security: A survey of challenges, solutions, and directions. Electronics, 14(7), 1394. https://doi.org/10.3390/electronics14071394

[6] Singh, N., Buyya, R., & Kim, H. (2024). Securing cloud-based Internet of Things: Challenges and mitigations. Sensors, 25(1), 79. https://doi.org/10.3390/s25010079

[7] Prakash, M. (2024). Why is Python important for cybersecurity? Retrieved from https://www.scaler.com/topics/cyber-security/cybersecurity-importance-for-python/

[8] Kumar, A., & Gutierrez, J.A. (2025). Impact of machine learning on intrusion detection systems for the protection of critical infrastructure. Information, 16(7), 515. https://doi.org/10.3390/info16070515

[9] Ilca, L.F., Lucian, O.P., & Balan, T.C. (2023). Enhancing cyber-resilience for small and medium-sized organizations with prescriptive malware analysis, detection and response. Sensors, 23(15), 6757. https://doi.org/10.3390/s23156757

[10] Coutinho, A.C., & de Araújo, L.V. (2025). MICRA: A modular intelligent cybersecurity response architecture with machine learning integration. Journal of Cybersecurity and Privacy, 5(3), 60. https://doi.org/10.3390/jcp5030060

[11] Dawood, M., Tu, S., Xiao, C., Alasmary, H., Waqas, M., & Rehman, S.U. (2023). Cyberattacks and security of cloud computing: A complete guideline. Symmetry, 15(11), 1981. https://doi.org/10.3390/sym15111981

[12] Jha, G. (2025). Introduction to cybersecurity. In Securing the enterprise. Apress, Berkeley, CA. https://doi.org/10.1007/979-8-8688-1654-3_1

[13] Sharma, T. (2025). An overview of Python’s popularity and versatility. Global Tech Council. Retrieved from https://www.globaltechcouncil.org/python/python-popularity-and-versatility/

[14] Kullberg, R. (2024). Python for cybersecurity: Key use cases and tools. Panther. Retrieved from https://panther.com/blog/python-for-cybersecurity-key-use-cases-and-tools

[15] Neville, M. (2025). Is Python right for your business in 2025? Strategic pros and cons. Softjourn. https://softjourn.com/insights/python-for-business