2024 International Conference on Computing, Networking and Communications (ICNC): Communications and Information Security Symposium

RMF-GPT — OpenAI GPT-3.5 LLM, Blockchain, NFT, Model Cards and OpenScap Enabled Intelligent RMF Automation System

Eranga Bandara*, Sachin Shetty*, Abdul Rahman†, Ravi Mukkamala*, Peter Foytik* Xueping Liang‡,
* {cmedawer, sshetty, mukka, pfoytik}@odu.edu
Old Dominion University, Norfolk, VA, USA
† abdulrahman@deloitte.com
Deloitte & Touche LLP
‡ xuliang@fiu.edu
Florida International University, USA

***Abstract*—The Risk Management Framework (RMF) provides a structured approach to managing risks to the confidentiality, integrity, and availability of information systems. However, automating the RMF process can be challenging due to various factors such as complexity, dealing with various standards (e.g. NIST SP 800-53), and supporting continuous Authority to Operate(ATO). In this research, we propose a solution to these issues through an end-to-end RMF automation system enabled by Custom-Trained "OpenAI GPT-3.5 LLM", blockchain, NFT, Model Cards, and OpenScap. The proposed system uses blockchain smart contracts to automate the vulnerability scanning and fixing process. Smart contracts interact with OpenScap API, which scans vulnerabilities on servers/nodes based on provided RMF checklists such as PCI DSS, NIST 800, STIG. The system then employs the custom-trained GPT-3.5 LLM(which powers the ChatGPT) to generates vulnerability fixing scripts (referred to as server hardening scripts) using Ansible/Puppet based on the identified vulnerabilities. Finally, the system runs these scripts to fix the vulnerabilities. This approach creates a fully automated RMF system that uses blockchain/smart contracts. All the system statuses, such as vulnerability and fixed status, are represented as NFT tokens with a customized NFT schema. The data provenance information is traced through Model Cards, which reduces the complexity of RMF automation and improves the capability of continuous ATOs. In this way, the proposed end-to-end RMF automation system enabled by GPT-3.5 LLM, Blockchain, NFT, Model Cards, and OpenScap addresses the challenges associated with RMF automation, providing a more efficient and effective way to manage the security of information systems.**

***Index Terms*—RMF, GPT, LLM, OpenAI, Blockchain, NFT, OpenScap**

I. Introduction

The Risk Management Framework (RMF) is a structured approach to managing risks to the confidentiality, integrity, and availability of information systems. The RMF provides a common language and framework for organizations to manage their information security risks, and it is widely used in both the public and private sectors [1]. The RMF process involves six steps: (1) Categorize the system, (2) Select and implement security controls, (3) Assess the security controls, (4) Authorize the system to operate, (5) Monitor the system and its environment, and (6) Conduct periodic reviews and updates.

One of the main challenges associated with the RMF is the complexity of the process. The process can be time-consuming and resource-intensive, particularly for large organizations with multiple information systems. Additionally, the RMF requires expertise in multiple domains, including security, risk management, and information technology, which can make it difficult to find qualified personnel [2]. Another challenge is the need to comply with various standards and regulations [3]. The RMF is often used to comply with government standards such as the National Institute of Standards and Technology (NIST) Special Publication (SP) 800-53 or the Department of Defense (DoD) Security Technical Implementation Guide (STIG). Compliance with these standards can be difficult, particularly as they are updated over time and require ongoing attention and effort to maintain. The complexity of control selection is another challenge associated with the RMF [4]. The RMF requires organizations to select and implement appropriate security controls based on their risk management strategy. However, the selection of controls can be a complex and time-consuming process, particularly for organizations with complex systems or those that are subject to multiple standards or regulations. Finally, supporting continuous Authorization to Operate (ATO) is a challenge in RMF. Organizations must continually monitor their systems and make necessary adjustments to maintain the appropriate level of security. This process can be difficult to manage, particularly for organizations with limited resources or those with complex information systems [5].

To address these challenges, we propose a novel end-to-end RMF automation system that integrates custom-trained OpenAI GPT-3.5 LLM [6], [7], blockchain, NFT [8], Model Cards [9], and OpenScap [10]. Our proposed system leverages blockchain smart contracts to automate the vulnerability scanning and remediation process. Through the interaction with the OpenScap API, the smart contracts facilitate comprehensive vulnerability scans on servers and nodes, adhering to

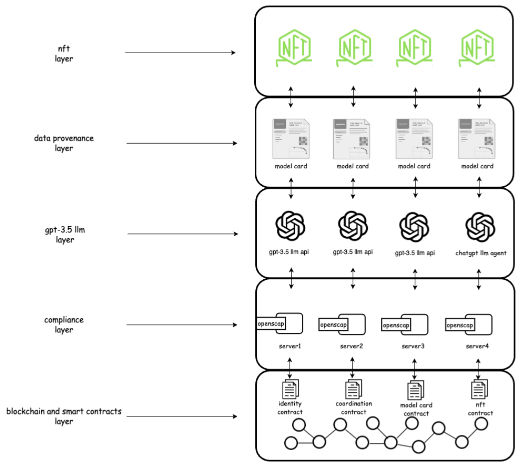


Fig. 1: Platform layered architecture.

established RMF checklists such as PCI DSS, NIST 800, and STIG. Upon identifying vulnerabilities, the system employs the custom-trained GPT-3.5 LLM and BabyAGI [11] to generate Ansible or Puppet-based scripts known as server hardening scripts, which are executed to address the identified vulnerabilities. This comprehensive approach creates a fully automated RMF system that utilizes blockchain and smart contracts. To represent critical system statuses, including vulnerability and fix status, our system employs customized NFT tokens with a specialized NFT schema. Moreover, the system incorporates Model Cards to trace and manage data provenance, effectively reducing the complexity of RMF automation and enhancing continuous ATO capabilities. By addressing the challenges associated with RMF automation, our proposed end-to-end system offers a streamlined and efficient approach to managing information system security. This paper has the following contributions:

1) Automate the end-to-end RMF process and support continuous ATOs with different RMF frameworks.
2) Utilizing the custom-trained GPT-3.5 LLM, the paper generates server-hardening scripts based on the identified vulnerabilities.
3) Proposed a model to represent system status updates as NFT tokens and designed an extensible NFT schema(k528) to represent the system statuses.
4) Encoded data provenance information of the RMF process into Model Cards and stored in the blockchain.

The rest of the paper is organized as follows. The design of the platform is discussed in Section 2. The platform's capabilities are discussed in Section 3. Section 4 consists of performance evaluation. Section 5 describes related work. Section 6 summarizes the proposed platform with recommendations for future work.

II. System Architecture

Figure 1 describes the architecture of the platform. The proposed platform is composed of four layers: 1) Blockchain/Smart Contract Layer, 2) Compliance Automation Layer, 3) GPT-3.5 LLM Layer 3) Data Provenance Layer, 3) NFT Layer, below is a brief description of each layer.

A. Blockchain and Smart Contract Layer

The blockchain and smart contract layer form the core/coordination layer of the platform, which can be deployed with multiple blockchain nodes. The blockchain ledger stores server identity information, data provenance information, and scanning results as NFTs [8]. The Blockchain smart contracts enable key features of the platform. These include the Identity contract, Compliance contract, Model Card contract, and NFT contract. The Identity contract handles identity management functions for servers and users, including identity registration, authentication, and access control. The servers in the system are equipped with OpenScap, a tool used for vulnerability scanning and fixing, which is managed by the RMF contract. The Model Card contract is responsible for data provenance, with the system's provenance information encoded into Model cards and stored in the blockchain ledger. The NFT contract manages all system statuses, including vulnerability and fixed status, as NFT tokens with a customized NFT schema k-528.

B. Compliance Automation Layer

The Compliance Automation Layer plays a vital role in automating the vulnerability scanning and fixing processes using OpenScap, effectively handling steps 4-7 of the RMF framework [1]. This layer ensures that all servers within the system are equipped with OpenScap and establishes seamless communication between the blockchain smart contracts and the OpenScap API. By utilizing RMF checklists like PCI DSS, NIST 800, and STIG [3], the system performs vulnerability scans on servers and nodes. It also enables the scheduling of periodic scans, allowing for regular assessment and mitigation of vulnerabilities, such as conducting scans on a weekly basis. In summary, the Compliance Automation Layer provides a comprehensive solution for automating the RMF process, simplifying security policy management, and ensuring ongoing compliance with the latest security standards.

C. GPT-3.5 LLM Layer

The GPT-3.5 LLM Layer utilizes the OpenAIs GPT-3.5 LLM, BabyAGI LLM Agent and LangChain to generate vulnerability fixing scripts, referred to as server-hardening scripts, based on identified vulnerabilities. These scripts are created as Ansible and Puppet playbooks, providing a standardized approach to remediation. Once generated, the blockchain smart contract executes these scripts on the respective servers, effectively addressing the identified vulnerabilities. The interaction among these components is detailed in Figure 2. The GPT-3.5 LLM has been specifically trained to generate vulnerability-fixing scripts. Blockchain smart contracts employ the GPT-3.5 LLM to create such scripts using the LangChain [12] API. The agent adheres to the architecture of BabyAGI and encompasses several crucial steps.

1) Task Creation Chain: For each identified threat, a list of tasks is generated to fix the associated vulnerability using the appropriate scripts.

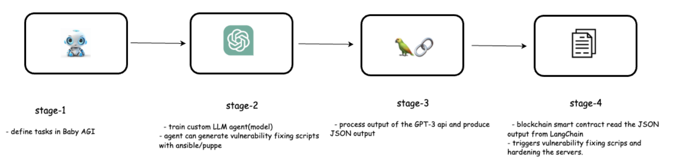

stage-1
- define tasks in Baby AGI
stage-2
- train custom LLM agent(model)
- agent can generate vulnerability fixing scripts with ansible/puppe
stage-3
- process output of the GPT-3 api and produce JSON output
stage-4
- blockchain smart contract read the JSON output from LangChain
- triggers vulnerability fixing scrips and hardening the servers.

Fig. 2: Server hardening flow with BabyAGI, GPT-3.5 LLM, LangChain and Smart Contracts.

2) Task Prioritization Chain: Following the principles of BabyAGI, this chain prioritizes the list of tasks based on their significance and impact.
3) Task Execution Chain: This chain executes the tasks, leveraging the generated scripts to effectively remediate the identified vulnerabilities.
4) Reflection: After executing a task, the Agent performs another scan to verify if the vulnerability has been successfully resolved.

The GPT-3.5 LLM Layer provides an intelligent and automated approach to vulnerability remediation, leveraging the capabilities of GPT-3.5 LLM and the principles of BabyAGI. By automating the process of generating and executing server hardening scripts, this layer contributes to the overall efficiency and effectiveness of the RMF automation system.

D. Data Provenance Layer

The data provenance layer is responsible for ensuring the integrity and traceability of all the actions and events that occur within the system. This is achieved using Model Cards, which record and store all the relevant information related to the vulnerability scanning and fixing process. These details may include the specific checklist or standard that was used for the scan, the version of the checklist or standard, and any additional details related to the scan itself, such as the type of scan performed (e.g., full, partial, or incremental), the scanning tool used, and any specific configurations or settings that were used during the scan. The model cards also capture information related to the vulnerabilities themselves, such as the severity level of the vulnerability, the affected component or system, and any associated risk factors [13]. Finally, the model cards record information related to the vulnerability fixing process, such as the start and end times of the fixing process, the individuals or teams responsible for fixing the vulnerabilities, and the specific scripts or configurations that were used to fix the vulnerabilities. By including all of this information in the model cards, the system is able to provide a comprehensive and detailed view of the entire vulnerability scanning and fixing process, which is critical for maintaining compliance, managing risk, and ensuring the overall security of the system.

The Model Card smart contract in the blockchain ledger is responsible for handling the functions related to model card generation, storage, and retrieval. It ensures that all the necessary information is captured in the model cards and that the data is properly structured and formatted [9]. By leveraging Model Cards and the blockchain ledger, the system is able to provide a high degree of transparency and accountability, making it easier to track and audit all the actions and events that occur within the system. This is especially important for organizations that need to comply with strict regulations or maintain continuous Authority to Operate (ATO) certifications.

E. NFT layer

The NFT object representing system vulnerabilities could include several pieces of information such as the name and description of the vulnerability, the severity level or score assigned to the vulnerability, and the date and time when the vulnerability was first detected. It could also include the type of vulnerability, such as SQL injection or cross-site scripting, and any associated CVE(Common Vulnerabilities and Exposures) identifiers [14]. Additionally, the NFT object may contain details about the vulnerability fix, such as the date and time it was fixed, the method used to fix it (e.g. patch installation or code modification), and the name of the person who fixed it. Other relevant information that could be included in the NFT object may include any scripts used to detect or fix the vulnerability, as well as any additional notes or comments related to the vulnerability or the fix. By including all this information in the NFT object, the system can maintain an immutable and transparent record of all vulnerabilities and their status throughout the entire vulnerability management lifecycle [8].

III. Platform Functionality

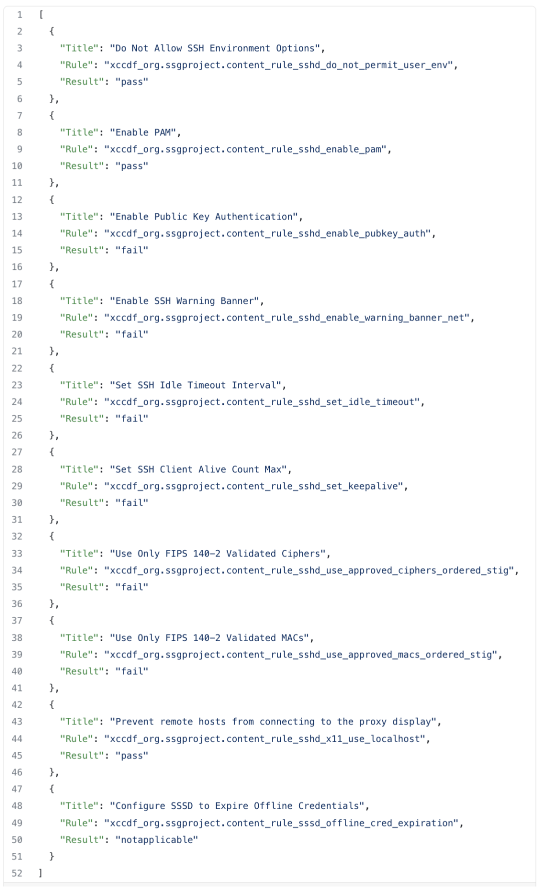
There are five main functionalities of the platform: 1) Peer Identity registration, 2) Vulnerability Scanning, 3) Vulnerability Fixing, 4) Continuous Monitoring and 5) Data provenance handling. This section goes into the specifics of these functions.

A. Peer Identity registration

The platform is designed to scan various servers that act as peers. Before conducting any scans, the servers' identities must be registered, which involves capturing the necessary information, such as the server name, IP address, MAC address, and other running operating system details. The Identity contract in the blockchain handles the registration process, creating an identity record in the blockchain storage. The compliance smart contracts can discover the peer by using this identity registry and perform vulnerability scanning and fixing vulnerabilities via OpenScap. Therefore, when onboarding servers into the platform, the OpenScap services must be installed to enable vulnerability scanning and fixing. For example, consider the scenario where Ubuntu 20.04 servers are onboarded into the platform. All the server identities, including IP addresses, server hostnames, and other related information, need to be registered in the blockchain ledger according to the identity registration process. To facilitate the identity registration process, we provide a web-based interface.

B. Vulnerability Scanning

The Compliance smart contract handles vulnerability scanning of servers onboarded into the platform via the identity registry. To execute vulnerability scanning, the Compliance contract uses the OpenScap APIs installed on the servers. For instance, let's consider scanning vulnerabilities in Ubuntu 20.04 servers based on the STIG guidelines and fixing them as per the STIG standard. The OpenScap contains SCAP documents related to Ubuntu 20.04 servers, such as "scap-security-guide-0.1.60/ssg-ubuntu2004-ds.xml" [15]. The SCAP document has variable providers that relate to different compliance standards such as STIG and CIS. The blockchain smart contract interacts with the OpenScap API in each server and instructs it to perform the scan according to the STIG profile. The scan generates a scan/audit report as an HTML file and an Asset Reporting Format (ARF) XML file. The report contains the server's STIG compliance score, which is based on the vulnerabilities found in the system, and it also includes detailed information about the vulnerabilities. Once the scan is completed, the scan results, identified vulnerabilities, scan scores, and other data are encoded into the NFT object and recorded in the blockchain ledger as shown in Figure 3. NFT token represents a specific server and its scanning results at a particular point in time. This allows for easy tracking and identification of each server's compliance status and vulnerabilities. Finally, NFT tokens can be easily transferred between parties, allowing for seamless sharing of compliance information between different entities. This can be particularly useful for regulatory compliance, where different parties may need to share compliance information with each other.



```json
[
  {
    "Title": "Do Not Allow SSH Environment Options",
    "Rule": "xccdf_org.ssgproject.content_rule_sshd_do_not_permit_user_env",
    "Result": "pass"
  },
  {
    "Title": "Enable PAM",
    "Rule": "xccdf_org.ssgproject.content_rule_sshd_enable_pam",
    "Result": "pass"
  },
  {
    "Title": "Enable Public Key Authentication",
    "Rule": "xccdf_org.ssgproject.content_rule_sshd_enable_pubkey_auth",
    "Result": "fail"
  },
  {
    "Title": "Enable SSH Warning Banner",
    "Rule": "xccdf_org.ssgproject.content_rule_sshd_enable_warning_banner_net",
    "Result": "fail"
  },
  {
    "Title": "Set SSH Idle Timeout Interval",
    "Rule": "xccdf_org.ssgproject.content_rule_sshd_set_idle_timeout",
    "Result": "fail"
  },
  {
    "Title": "Set SSH Client Alive Count Max",
    "Rule": "xccdf_org.ssgproject.content_rule_sshd_set_keepalive",
    "Result": "fail"
  },
  {
    "Title": "Use Only FIPS 140-2 Validated Ciphers",
    "Rule": "xccdf_org.ssgproject.content_rule_sshd_use_approved_ciphers_ordered_stig",
    "Result": "fail"
  },
  {
    "Title": "Use Only FIPS 140-2 Validated MACs",
    "Rule": "xccdf_org.ssgproject.content_rule_sshd_use_approved_macs_ordered_stig",
    "Result": "fail"
  },
  {
    "Title": "Prevent remote hosts from connecting to the proxy display",
    "Rule": "xccdf_org.ssgproject.content_rule_sshd_x11_use_localhost",
    "Result": "pass"
  },
  {
    "Title": "Configure SSSD to Expire Offline Credentials",
    "Rule": "xccdf_org.ssgproject.content_rule_sssd_offline_cred_expiration",
    "Result": "notapplicable"
  }
]
```

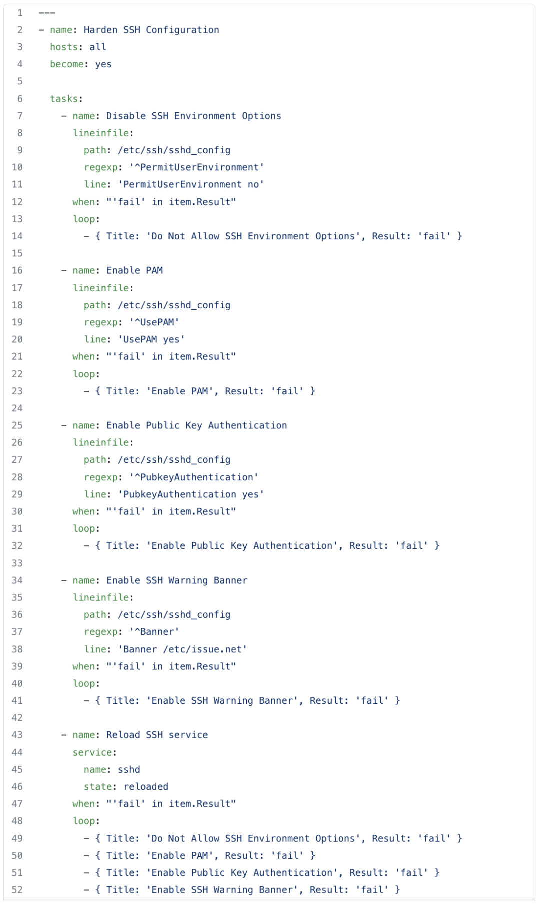
Fig. 3: Vulnerability status(scan result) NFT object.

C. Vulnerability Fixing

After generating the vulnerability reports, the system proceeds to strengthen the server's security through the creation of a SCAP Security Guide (SSG). In our proposed solution, blockchain smart contracts seamlessly interact with the OpenScap API to generate an SSG guide specifically tailored for STIG compliance. Moreover, our system offers automated server hardening capabilities based on the identified vulnerabilities. To achieve this, the blockchain smart contract leverages the custom-trained GPT-3.5 LLM to automatically produce an Ansible playbook or bash script, aligning with security compliance standards such as STIG. The Ansible playbook is dynamically generated by considering the vulnerabilities identified within the system and adhering to the STIG compliance requirements, Figure 4. Once the server hardening playbook is ready, the smart contract triggers its execution on the respective server, effectively implementing the necessary hardening measures [10]. Once the vulnerabilities have been successfully addressed, the system can re-run the scanning process, recording the updated vulnerability status as an NFT object. This unique approach allows for the representation of the server's scanning result snapshot as an NFT token, facilitating easy tracking and monitoring of the server's security status over time.

D. Continuous Monitoring

The platform offers a robust capability for scheduled and continuous vulnerability scanning and remediation using blockchain smart contracts. This feature empowers organizations to regularly assess their servers for vulnerabilities and automatically address any identified issues to uphold their security compliance posture. For instance, organizations can establish recurring scans on a weekly or monthly basis to ensure the ongoing security and compliance of their servers [5]. The smart contracts are designed to autonomously trigger the scans at the scheduled intervals, generating comprehensive vulnerability reports. These reports encompass critical details, including the nature of the identified vulnerabilities, their severity level, and recommended remediation measures. Leveraging the GPT-3.5 LLM capabilities, the system utilizes the vulnerability reports to generate tailored Ansible playbooks or bash scripts, specifically crafted to address the identified issues effectively. The smart contracts orchestrate the execution of these playbooks on the respective servers, seamlessly and automatically resolving the vulnerabilities. Once the remediation actions have been applied, the API initiates another round of scanning to capture the updated vulnerability status. This refreshed status is then recorded as an NFT object within the blockchain ledger, facilitating transparent and immutable tracking of the server's security



```yaml
---
- name: Harden SSH Configuration
  hosts: all
  become: yes

  tasks:
    - name: Disable SSH Environment Options
      lineinfile:
        path: /etc/ssh/sshd_config
        regexp: '^PermitUserEnvironment'
        line: 'PermitUserEnvironment no'
      when: "'fail' in item.Result"
      loop:
        - { Title: 'Do Not Allow SSH Environment Options', Result: 'fail' }

    - name: Enable PAM
      lineinfile:
        path: /etc/ssh/sshd_config
        regexp: '^UsePAM'
        line: 'UsePAM yes'
      when: "'fail' in item.Result"
      loop:
        - { Title: 'Enable PAM', Result: 'fail' }

    - name: Enable Public Key Authentication
      lineinfile:
        path: /etc/ssh/sshd_config
        regexp: '^PubkeyAuthentication'
        line: 'PubkeyAuthentication yes'
      when: "'fail' in item.Result"
      loop:
        - { Title: 'Enable Public Key Authentication', Result: 'fail' }

    - name: Enable SSH Warning Banner
      lineinfile:
        path: /etc/ssh/sshd_config
        regexp: '^Banner'
        line: 'Banner /etc/issue.net'
      when: "'fail' in item.Result"
      loop:
        - { Title: 'Enable SSH Warning Banner', Result: 'fail' }

    - name: Reload SSH service
      service:
        name: sshd
        state: reloaded
      when: "'fail' in item.Result"
      loop:
        - { Title: 'Do Not Allow SSH Environment Options', Result: 'fail' }
        - { Title: 'Enable PAM', Result: 'fail' }
        - { Title: 'Enable Public Key Authentication', Result: 'fail' }
        - { Title: 'Enable SSH Warning Banner', Result: 'fail' }
```

Fig. 4: Ansbile playbook generated by GPT-3.5 LLM.

posture. Through this continuous vulnerability scanning and remediation process, the platform ensures that servers remain up-to-date with the latest security patches and consistently align with pertinent regulations and standards. By leveraging blockchain technology and smart contracts, organizations can proactively safeguard their systems against emerging threats, fortify their security resilience, and maintain ongoing compliance.

E. Data provenance handling

The platform also records data provenance information for scanning and vulnerability fixing activities into model cards. A model card is a standardized format used for documenting machine learning models' performance and associated metadata. In the context of our platform, the model cards are used to document the scanning and vulnerability fixing processes and their outcomes, including the identified vulnerabilities, the scan reports, and the fixes applied. The data provenance information is recorded in the model cards to provide transparency and traceability of the processes and ensure compliance with regulatory requirements. The model cards also contain information about the scanning and fixing algorithms used, their accuracy, and their limitations. This information can be useful for auditors, regulators, and stakeholders to assess the effectiveness of the scanning and fixing processes and the

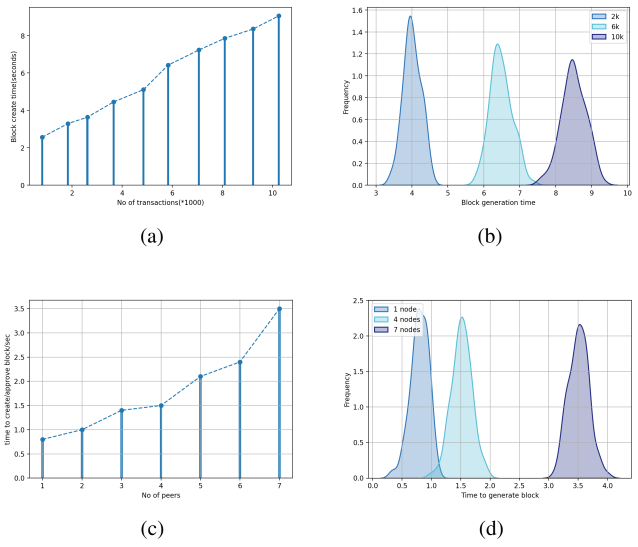


Fig. 5: Block generation time.

overall security posture of the system. The model cards are stored in the blockchain ledger, ensuring their immutability and tamper-proof nature.

IV. Implementation and Evaluation

The platform's design incorporates the Rahasak blockchain ledger [16], [17] and employs the Aplos smart contract platform [18] to implement essential functions. We have trained OpenAI's GPT-3.5-turbo model to generate vulnerability fixing scripts. The LLM training and querying were performed through OpenAI APIs, BabyAGI and LangChain [11], [12]. An evaluation of the platform's performance across varying numbers of blockchain peers is summarized below.

The block generation time is a critical component because it determines the system's effectiveness. First, information on block creation time is compared to the number of transactions in the block as demonstrated in Figure 5(a). The average block generation time with different transaction sets in a block is discussed in Figure 5(b). Next, the block generation time was measured in different cases where the number of blockchain peers in the cluster (up to 7) was changed. When adding new peers to the network, the block creation time is shown in Figure 5(c). Figure 5(d) depicts the average block production time when the network has a variable number of blockchain peers.

V. Related Work

Various researchers tried to facilitate RMF functions with blockchain. The key elements and architecture of these research initiatives are outlined in this section. Table I summarizes the contrast between these projects and the proposed platform.

"Continuous Cybersecurity Management Through Blockchain Technology" [3] introduces a novel approach utilizing blockchain technology to address delays in product

TABLE I: RMF automation platform comparison

Platform	Centralized/ Distributed	Blockchain Support	Running Blockchain	Supported RMF Frameworks	Data Provenance Support	NFT Support	Continuous ATO Support	AI Integration
RMF-GPT	Distributed	✓	Moose	PCI DSS, NIST 800, STIG	✓	✓	✓	✓
Contineous RMF [3]	Distributed	✓	N/A	NIST	✗	✗	✗	✗
Perspective RMF [19]	Distributed	✓	N/A	N/A	✗	✗	✗	✗
Letstrace [13]	Distributed	✓	Rahasak	N/A	✓	✗	✗	✗
Vind [14]	Distributed	✓	Rahasak	N/A	✓	✗	✗	✗
SmartGrid-RMF [20]	Distributed	✗	N/A	NIST	✓	✗	✓	✗

release cycles and enhance security and functionality. The paper ”Perspectives on risks and standards that affect the requirements engineering of blockchain technology” [19] explores the potential of blockchain technology to revolutionize business transactions by introducing a trust model based on algorithms. ”Letstrace” [13] presents a blockchain-based cyber supply chain provenance platform that integrates TUF and In-ToTo frameworks, verifying software updates and enhancing supply chain security. In ”vind” [14], a blockchain-based cyber supply chain provenance platform is proposed to address vulnerabilities in the Energy Delivery Systems supply chain. The paper ”Blockchain Technology for Smart Grids: Decentralized NIST Conceptual Model” [20] evaluates the impact of blockchain on decentralizing smart grids using the NIST conceptual model.

VI. Conclusions and Future Work

In conclusion, our research proposes an end-to-end RMF automation system enabled by GPT-3.5 LLM, Blockchain, NFT, Model Cards, and OpenScap to address the challenges associated with RMF automation. The system uses blockchain smart contracts to automate vulnerability scanning using OpenScap. Then system will generates server hardening scripts to fix identified vulnerabilities using custom-trained GPT-3.5 LLM. All system statuses are represented as NFT tokens with a customized NFT schema, and data provenance information is traced through Model Cards. This approach creates a fully automated RMF system that uses blockchain/smart contracts, providing a more efficient and effective way to manage the security of information systems and improving the capability of continuous ATOs. The proposed solution offers significant benefits to information system security, and future research can explore the potential of this technology further. In the future, we plan to extend the proposed solution to cover other compliance checklists and integrate it with a federated learning system to enable more advanced threat analysis.

Acknowledgements

This work is supported in part by DoD Center of Excellence in AI and Machine Learning (CoE-AIML) under Contract Number W911NF-20-2-0277 with the U.S. Army Research Laboratory, National Science Foundation under Grant No. 2219742 and Grant No. 2131001

References

[1] R. Ross, “Managing enterprise security risk with nist standards,” *Computer*, vol. 40, no. 8, pp. 88–91, 2007.

[2] ——, “Managing enterprise security risk with nist standards,” *Computer*, vol. 40, no. 8, pp. 88–91, 2007.

[3] J. White and C. Daniels, “Continuous cybersecurity management through blockchain technology,” in *2019 IEEE Technology & Engineering Management Conference (TEMSCON)*. IEEE, 2019, pp. 1–5.

[4] B. R. Williams and J. Adamson, *PCI Compliance: Understand and implement effective PCI data security standard compliance*. CRC Press, 2022.

[5] T. Chick and T. Scanlon, “Risk management framework (rmf) and authority to operate (ato),” 2023.

[6] I. L. Alberts, L. Mercolli, T. Pyka, G. Prenosil, K. Shi, A. Rominger, and A. Afshar-Oromieh, “Large language models (llm) and chatgpt: what will the impact on nuclear medicine be?” *European journal of nuclear medicine and molecular imaging*, pp. 1–4, 2023.

[7] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell *et al.*, “Language models are few-shot learners,” *Advances in neural information processing systems*, vol. 33, pp. 1877–1901, 2020.

[8] R. Chohan and J. Paschen, “What marketers need to know about non-fungible tokens (nfts),” *Business Horizons*, 2021.

[9] A. Wadhwani and P. Jain, “Machine learning model cards transparency review: Using model card toolkit,” in *2020 IEEE Pune Section International Conference (PuneCon)*. IEEE, 2020, pp. 133–137.

[10] J. A. Webb, M. W. Henderson, and M. L. Webb, “An open source approach to automating surveillance and compliance of automatic test systems,” in *2019 IEEE AUTOTESTCON*. IEEE, 2019, pp. 1–8.

[11] G. T. Zhao, “A comprehensive and hands-on guide to autonomous agents with gpt.”

[12] S. Ott, K. Hebenstreit, V. Liévin, C. E. Hother, M. Moradi, M. Mayrhauser, R. Praas, O. Winther, and M. Samwald, “Thoughtsource: A central hub for large language model reasoning data,” *arXiv preprint arXiv:2301.11596*, 2023.

[13] E. Bandara, S. Shetty, A. Rahman, and R. Mukkamala, “Let’strace—blockchain, federated learning and tuf/in-toto enabled cyber supply chain provenance platform,” in *MILCOM 2021-2021 IEEE Military Communications Conference (MILCOM)*. IEEE, 2021, pp. 470–476.

[14] E. Bandara, S. Shetty, D. Tosh, and X. Liang, “Vind: A blockchain-enabled supply chain provenance framework for energy delivery systems,” *Frontiers in Blockchain*, p. 28, 2021.

[15] M. Preisler and M. Haicman, “Security automation for containers and {VMs} with {OpenSCAP},” 2017.

[16] E. Bandara, X. Liang, P. Foytik, S. Shetty, N. Ranasinghe, and K. De Zoysa, “Rahasak-scalable blockchain architecture for enterprise applications,” *Journal of Systems Architecture*, p. 102061, 2021.

[17] E. Bandara, D. Tosh, P. Foytik, S. Shetty, N. Ranasinghe, and K. De Zoysa, “Tikiri-towards a lightweight blockchain for iot,” *Future Generation Computer Systems*, 2021.

[18] E. Bandara, W. K. NG, K. De Zoysa, and N. Ranasinghe, “Aplos: Smart contracts made smart,” *BlockSys’2019*, 2019.

[19] N. Drljevic, D. A. Aranda, and V. Stantchev, “Perspectives on risks and standards that affect the requirements engineering of blockchain technology,” *Computer Standards & Interfaces*, vol. 69, p. 103409, 2020.

[20] A. Aderibole, A. Aljarwan, M. H. U. Rehman, H. H. Zeineldin, T. Mezher, K. Salah, E. Damiani, and D. Svetinovic, “Blockchain technology for smart grids: Decentralized nist conceptual model,” *IEEE Access*, vol. 8, pp. 43 177–43 190, 2020.