ChatGPT Created Threat Model

Priya Joy Kaviyil
Atlantic Technological University
Letterkenny, Ireland
L00171183@atu.ie

Danny McFadden
Atlantic Technological University
Letterkenny, Ireland
Danny.McFadden@atu.ie

Ruth Lennon
Atlantic Technological University
Letterkenny, Ireland
Ruth.Lennon@atu.ie

***Abstract*— This research investigates the integration of ChatGPT, a powerful language model, into the threat modeling process to enhance efficiency in addressing high-priority threats. By leveraging capabilities of ChatGPT, the study explores its potential in identifying and mitigating vulnerabilities, contributing to improved threat modeling. However, it emphasizes that ChatGPT should not replace human expertise but rather be used in conjunction with traditional methodologies. The research points out the importance of a comprehensive approach involving industry guidelines and expert consultations to develop robust threat models. The findings highlight the value of ChatGPT as a supportive tool for efficient threat modeling while underscoring the necessity of human judgment and expertise.**

Keywords—Threat Modelling, ChatGPT, Artificial Intelligence, security, risks, mitigations

I. Introduction

Threat modeling is on the rise as standards emphasize its importance, resulting in early vulnerability detection and improved long-term product quality [1]. Threat modeling have proven to be highly efficient, however they don’t scale well to keep up with the ever-evolving computing and threat environment [2]. With the increasing digitalization of business operations, addressing all high-priority threats using traditional methods has become exceedingly time-consuming, resulting in numerous unattended vulnerabilities.

ChatGPT is an AI chatbot that has gained immense popularity and has become a sensation among users since its release in late 2022. It is being utilized for a wide range of applications, including coding assistance, code reviews, policy drafting, blog writing, and numerous other use cases [3].

In this research, we conducted an investigation into the utilization of ChatGPT for threat modeling. Our objective is to determine if leveraging the capabilities of ChatGPT can help overcome the efficiency challenges faced by traditional threat modeling approaches in the present computing and threat landscape. By integrating ChatGPT into the threat modeling process, we aim to explore whether it can offer a viable solution to the time-consuming nature of addressing an organization's high-priority threats. This research seeks to assess the potential of ChatGPT in effectively identifying and mitigating vulnerabilities, ultimately contributing to enhanced threat modeling. And for executing this approach we created a small microservice system model connected to MongoDB and hosted on AWS cloud.

II. Background Reading

A. *Threat Modelling*

Microsoft introduced the concept of threat modelling at the turn of the century [4]. The findings presented in the book authored by Swiderski and Snyder [5] were officially documented and subsequently integrated as an integral part of the initial version of the Microsoft Security Development Lifecycle (SDL) [6]. Threat modelling is a crucial process in identifying, communicating, and managing security weaknesses; provides a deeper understanding of critical aspects of the system and enables organizations to identify vulnerabilities and potential security threats [7]. During the threat modelling process, developers and security experts analyse the application's architecture, data flows, and access controls to identify potential threats and security vulnerabilities; this helps developers to realize the security consequences of their design, code, and configuration choices, and to implement effective security measures to mitigate the identified threats [7].

Threat modelling encompasses a range of approaches, including conceptual frameworks and practical tools, with the goal of understanding the complexities of a system and identifying potential threats to it, as defined by Myagmar, Lee, and Yurcik [8]. Shostack [9] suggested that the process of threat modelling commonly involves the creation of two models: one that represents the system to be developed and another that illustrates the actual threats to the system along with their corresponding mitigations. For this research on creating threat model using ChatGPT, we are developing two models, a system model that portrays the microservice connected to MongoDB and hosted in the AWS cloud, alongside a threat model that highlights the identified threats and the corresponding measures to mitigate them with the assistance of ChatGPT.

There is a wide range of threat modelling methodologies that companies can make use of, as each is a unique approach and provides varied benefits. Among these, the most common are STRIDE, OCTAVE, TRIKE AND PASTA [10]. Threat modelling methodologies aids in generating a system abstraction and offering analyses of potential attackers, including their objectives and techniques. Moreover, it provides valuable insights on potential vulnerabilities and threats that may arise in the future. These are some of the best methodologies used, which have unique methods and frameworks to identify, analyse, measure, and sort threats [11].

With regards to tools, Microsoft's Threat Modelling Tool (MS-TMT) [13] is a widely used freely available tool for threat modelling. There are other alternatives in the market, one such tool is OWASP Threat Dragon [14], which supports Windows, MacOS and Linux and a web app. SPARTA [16] is another tool in the realm of threat modelling that enhances the STRIDE approach [15] by incorporating Data Flow Diagrams (DFDs) to establish explicit connections between identified threats and corresponding countermeasures. SPARTA goes beyond mere threat identification by employing simulations to estimate the vulnerability of the solution, taking into consideration the capabilities of various attacker types.

To initiate the process of threat modelling, a diagram is often created in every methodology that outlines the system's architecture, components, trust zones, and authentication flows [17], [18]. The inclusion of data flows in a diagram can prove to be extremely advantageous, as it provides a clear representation of how information is received and transmitted by the system, how it is altered, and where it is stored [17]. The primary goal of a data flow diagram (DFD) is to provide an overview of the system's scope and boundaries as a whole and a comprehensive analysis of the system's security posture. For this paper, in the initial stage we are utilizing DFDs for the system model and the threat model to carry out the process of threat modelling the microservice system.

B. *ChatGPT*

ChatGPT, created by OpenAI, was launched on November 30, 2022, as an AI-powered natural language processing tool that enables you to engage in human-like conversations and offers a wide range of capabilities and it is available for free, allowing you to ask unlimited questions and engage in conversations [19]. The architecture powering ChatGPT is based on the Generative Pre-trained Transformer (GPT) developed by OpenAI [19]. As stated by OpenAI [20], ChatGPT specifically utilizes a fine-tuned version from the GPT-3.5 series. To access ChatGPT, you can easily visit chat.openai.com [21], create an OpenAI account, and begin conversing. According to a recent analysis conducted by UBS [22], ChatGPT has achieved unprecedented growth as the fastest-growing app in history. The analysis suggests that within just two months of its launch, ChatGPT had already amassed an impressive user base of 100 million active users in January.

ChatGPT serves as a versatile tool capable of assisting with an unlimited range of projects, spanning various domains such as software development, writing, and translations [19]. With its ability to generate responses to prompts we can ask any burning questions to ChatGPT; it has the potential to become a significant tool for content generation, surpassing traditional search engines. It can assist in diverse tasks such as writing essays, summarizing books, coding and debugging, performing calculations, resume compilation, translating information, and much more.

In this research paper, we have explored an innovative approach to threat modelling that incorporates a combination of these different threat modelling methodologies and tools by leveraging the capabilities of ChatGPT; evaluate the effectiveness of this novel approach and its potential impact.

III. Methodology

A. *Objective*

Upon reviewing the background study, several methodological approaches have been identified that can be effectively employed for the purpose of conducting threat modelling. The aim is to create a methodology to model threats using ChatGPT. Creating a threat model using ChatGPT relies on an artificial intelligence language model that can analyse vast amounts of data and generate insights based on that analysis. This method is unlike the conventional methodologies which typically rely on manual processes that involve subject matter experts, stakeholders, and other key personnel to identify and assess potential threats to a system. The objective for this paper is creating a threat model for a microservice system connected to a MongoDB instance and hosted in AWS using ChatGPT which would assist in identifying potential threats and vulnerabilities that could impact the system's security. And by analysing the system and its associated risks, developing a comprehensive threat model that outlines specific measures to mitigate the identified threats.

B. *Approach*

This section presents the methodology used in the paper to establish a structured threat modelling approach for a microservice system. Firstly, a Data Flow Diagram (DFD) is created for the system model (Fig.1), which outlines the various dataflows and services used in the system. The system model serves as the input model for asking ChatGPT to create the threat model, which is then used to generate a threat model for the system. Based on the outputs provided by ChatGPT, a DFD for the threat model is created (Fig. 3), which outlines the potential threats that could impact the system's security. To address these threats, solutions are asked to ChatGPT which will be implemented to mitigate the identified threats.

1) System Model: A model is created for a spring boot microservice built which is connected to a mongo dB database. Both the microservice and database are hosted in AWS EC2 virtual machine using the EC2 and S3 bucket AWS services. Created an AWS Linux EC2 instance from the AWS Management Console by launching a new instance and configured its settings, including selecting the desired Linux AMI. Security groups were set up to allow incoming connections on MongoDB and microservice ports. Accessed the EC2 instance from local terminal using a secret key pair by setting the key file's permissions and using SSH. Downloaded and installed MongoDB on the EC2 instance, updated its configuration to accept remote connections, and created a MongoDB user. Installed MongoDB Compass on the local machine and connected to the MongoDB instance on the EC2 instance. Created a Spring Boot microservice with IntelliJ, built and compiled it to generate a JAR file, and created an S3 bucket in AWS to store the JAR. Jar file was uploaded to the S3 bucket and copied to the EC2 instance. Successfully started the microservice application on the EC2 instance and tested its endpoints using a tool like Postman, specifying the EC2 instance's public IP address or DNS name.

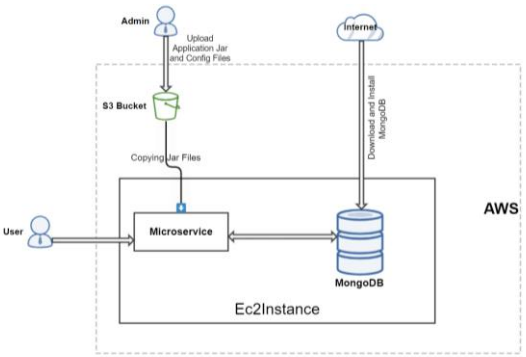
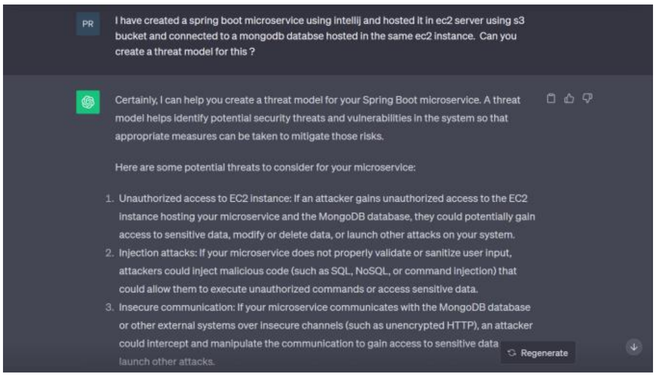


Fig. 1. Microservice System Model

2) Threat Model: Asked ChatGPT to create a threat model for the microservice system model created. ChatGPT provided a list of potential threats (Fig.2) and mitigation measures (Fig.4). A threat model has been developed solely based on the provided information (Fig.3). According to ChatGPT [23], here are the potential threats to consider for the system:

- Unauthorized access to EC2 instance and MongoDB database.
- Injection attacks, such as SQL, NoSQL, or command injection.
- Insecure communication channels, leading to interception or manipulation of data.
- Insufficient authentication and authorization mechanisms.
- Denial of service (DoS) attacks, causing unavailability or unresponsiveness.
- Weak passwords or credentials.



I have created a spring boot microservice using intellij and hosted it in ec2 server using s3 bucket and connected to a mongodb databse hosted in the same ec2 instance. Can you create a threat model for this ?

Certainly, I can help you create a threat model for your Spring Boot microservice. A threat model helps identify potential security threats and vulnerabilities in the system so that appropriate measures can be taken to mitigate those risks.

Here are some potential threats to consider for your microservice:

1. Unauthorized access to EC2 instance: If an attacker gains unauthorized access to the EC2 instance hosting your microservice and the MongoDB database, they could potentially gain access to sensitive data, modify or delete data, or launch other attacks on your system.
2. Injection attacks: If your microservice does not properly validate or sanitize user input, attackers could inject malicious code (such as SQL, NoSQL, or command injection) that could allow them to execute unauthorized commands or access sensitive data.
3. Insecure communication: If your microservice communicates with the MongoDB database or other external systems over insecure channels (such as unencrypted HTTP), an attacker could intercept and manipulate the communication to gain access to sensitive data launch other attacks.

Fig. 2. Asking ChatGPT to create threat model [23]

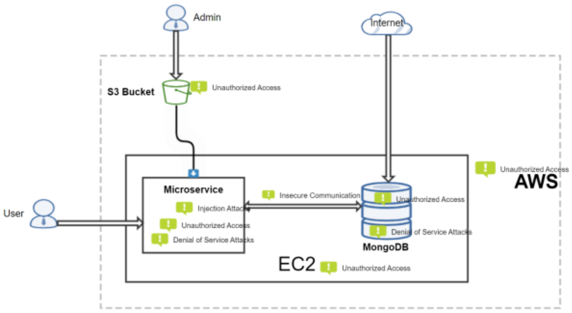
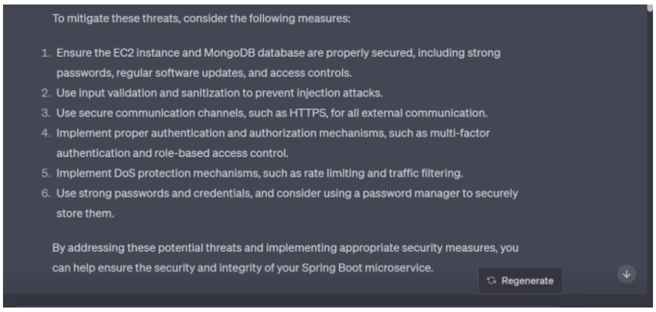


Fig. 3. ChatGPT Created Threat Model


To mitigate these threats, consider the following measures:

1. Ensure the EC2 instance and MongoDB database are properly secured, including strong passwords, regular software updates, and access controls.
2. Use input validation and sanitization to prevent injection attacks.
3. Use secure communication channels, such as HTTPS, for all external communication.
4. Implement proper authentication and authorization mechanisms, such as multi-factor authentication and role-based access control.
5. Implement DoS protection mechanisms, such as rate limiting and traffic filtering.
6. Use strong passwords and credentials, and consider using a password manager to securely store them.

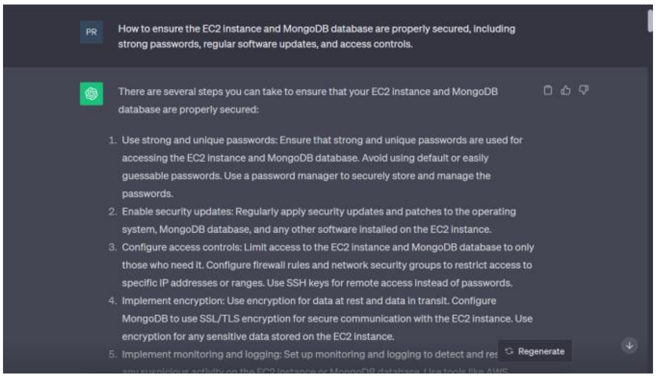
By addressing these potential threats and implementing appropriate security measures, you can help ensure the security and integrity of your Spring Boot microservice.

Fig.4: Mitigations [23]

3) Threat Mitigations: Corresponding mitigation measures for the potential threats were provided by ChatGPT (Fig.4). According to ChatGPT [24] [25] [26] [27], here are the key points for ensuring the security of Spring Boot microservice, EC2 instance, and MongoDB database:

For the EC2 instance and MongoDB database (Fig. 5):

- Use strong and unique passwords.
- Regularly apply security updates and patches.
- Configure access controls and firewall rules.
- Implement encryption for data at rest and in transit.
- Set up monitoring and logging.
- Take regular backups of the database



How to ensure the EC2 instance and MongoDB database are properly secured, including strong passwords, regular software updates, and access controls.

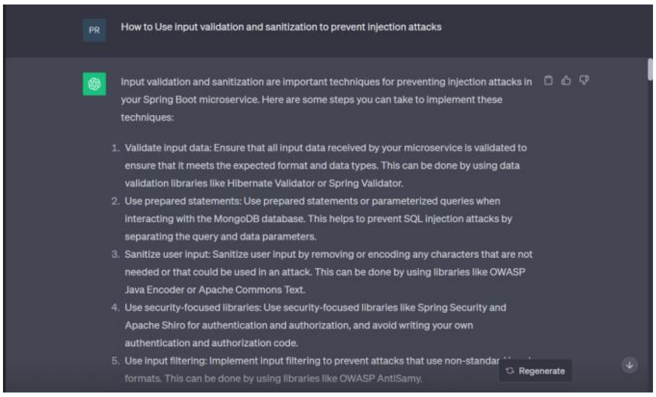
There are several steps you can take to ensure that your EC2 instance and MongoDB database are properly secured:

1. Use strong and unique passwords: Ensure that strong and unique passwords are used for accessing the EC2 instance and MongoDB database. Avoid using default or easily guessable passwords. Use a password manager to securely store and manage the passwords.
2. Enable security updates: Regularly apply security updates and patches to the operating system, MongoDB database, and any other software installed on the EC2 instance.
3. Configure access controls: Limit access to the EC2 instance and MongoDB database to only those who need it. Configure firewall rules and network security groups to restrict access to specific IP addresses or ranges. Use SSH keys for remote access instead of passwords.
4. Implement encryption: Use encryption for data at rest and data in transit. Configure MongoDB to use SSL/TLS encryption for secure communication with the EC2 instance. Use encryption for any sensitive data stored on the EC2 instance.
5. Implement monitoring and logging: Set up monitoring and logging to detect and res any suspicious activity on the EC2 instance or MongoDB database. Use tools like AWS

Fig. 5. ChatGPT - Ensuring Security of EC2 Instance and MongoDB [24]

For preventing injection attacks (Fig. 6):

- Validate input data to ensure the expected format and data types.
- Use prepared statements or parameterized queries.
- Sanitize user input by removing or encoding unnecessary characters.
- Utilize security-focused libraries for authentication and authorization.
- Implement input filtering and length limitations



PR How to Use input validation and sanitization to prevent injection attacks

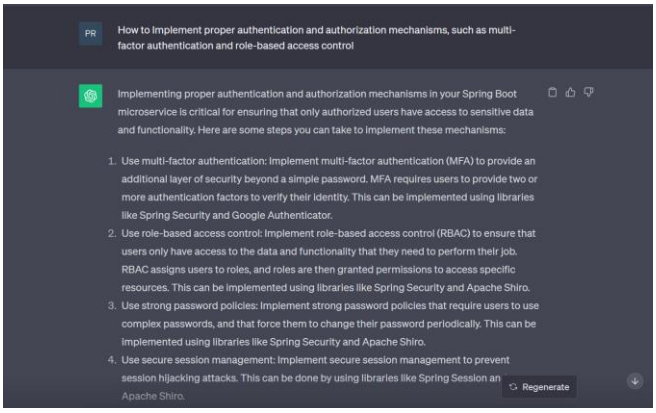
Input validation and sanitization are important techniques for preventing injection attacks in your Spring Boot microservice. Here are some steps you can take to implement these techniques:

1. Validate input data: Ensure that all input data received by your microservice is validated to ensure that it meets the expected format and data types. This can be done by using data validation libraries like Hibernate Validator or Spring Validator.
2. Use prepared statements: Use prepared statements or parameterized queries when interacting with the MongoDB database. This helps to prevent SQL injection attacks by separating the query and data parameters.
3. Sanitize user input: Sanitize user input by removing or encoding any characters that are not needed or that could be used in an attack. This can be done by using libraries like OWASP Java Encoder or Apache Commons Text.
4. Use security-focused libraries: Use security-focused libraries like Spring Security and Apache Shiro for authentication and authorization, and avoid writing your own authentication and authorization code.
5. Use input filtering: Implement input filtering to prevent attacks that use non-standar formats. This can be done by using libraries like OWASP AntiSamy.

Fig. 6. ChatGPT - Input Validation and Sanitization to prevent injection attacks [25]

For implementing authentication and authorization mechanisms (Fig. 7):

- Use multi-factor authentication, such as MFA.
- Implement role-based access control (RBAC).
- Enforce strong password policies and secure session management.
- Implement API gateway security to control access



PR How to implement proper authentication and authorization mechanisms, such as multi-factor authentication and role-based access control

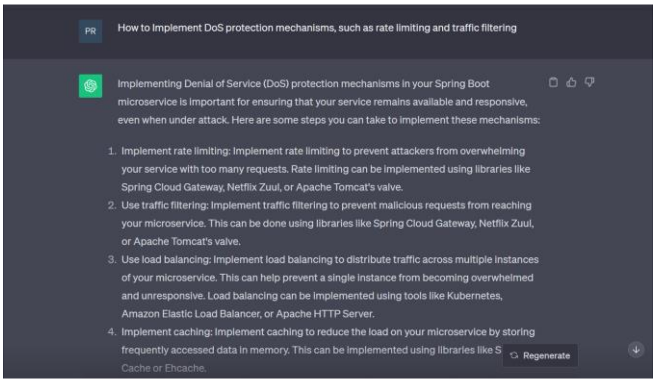
Implementing proper authentication and authorization mechanisms in your Spring Boot microservice is critical for ensuring that only authorized users have access to sensitive data and functionality. Here are some steps you can take to implement these mechanisms:

1. Use multi-factor authentication: Implement multi-factor authentication (MFA) to provide an additional layer of security beyond a simple password. MFA requires users to provide two or more authentication factors to verify their identity. This can be implemented using libraries like Spring Security and Google Authenticator.
2. Use role-based access control: Implement role-based access control (RBAC) to ensure that users only have access to the data and functionality that they need to perform their job. RBAC assigns users to roles, and roles are then granted permissions to access specific resources. This can be implemented using libraries like Spring Security and Apache Shiro.
3. Use strong password policies: Implement strong password policies that require users to use complex passwords, and that force them to change their password periodically. This can be implemented using libraries like Spring Security and Apache Shiro.
4. Use secure session management: Implement secure session management to prevent session hijacking attacks. This can be done by using libraries like Spring Session an Apache Shiro.

Fig. 7. ChatGPT - Implement authentication and authorization mechanisms [26]

For DoS protection (Fig. 8):

- Implement rate limiting and traffic filtering.
- Utilize load balancing and caching mechanisms.
- Consider using cloud-based DoS protection services.



PR How to Implement DoS protection mechanisms, such as rate limiting and traffic filtering

Implementing Denial of Service (DoS) protection mechanisms in your Spring Boot microservice is important for ensuring that your service remains available and responsive, even when under attack. Here are some steps you can take to implement these mechanisms:

1. Implement rate limiting: Implement rate limiting to prevent attackers from overwhelming your service with too many requests. Rate limiting can be implemented using libraries like Spring Cloud Gateway, Netflix Zuul, or Apache Tomcat's valve.
2. Use traffic filtering: Implement traffic filtering to prevent malicious requests from reaching your microservice. This can be done using libraries like Spring Cloud Gateway, Netflix Zuul, or Apache Tomcat's valve.
3. Use load balancing: Implement load balancing to distribute traffic across multiple instances of your microservice. This can help prevent a single instance from becoming overwhelmed and unresponsive. Load balancing can be implemented using tools like Kubernetes, Amazon Elastic Load Balancer, or Apache HTTP Server.
4. Implement caching: Implement caching to reduce the load on your microservice by storing frequently accessed data in memory. This can be implemented using libraries like S Cache or Ehcache.

Fig. 8. ChatGPT - Implement DoS protection mechanisms [27]

4) *Implementation:* The recommended mitigation measures provided by ChatGPT for securing AWS, EC2 instance, Spring Boot microservice, and MongoDB were implemented successfully. This included incorporating effective Denial of Service (DoS) protection mechanisms, implementing robust authentication and authorization mechanisms, performing thorough input validation and sanitization to prevent injection attacks, and ensuring the overall security of both the EC2 instance and MongoDB database inside the AWS cloud. The measures included employing IAM User access control and MFA codes for the EC2 instance, implementing Security Groups and SSH connections (such as using Putty) for secure access, and utilizing CloudWatch for monitoring EC2 log (Fig. 9). For MongoDB, the security measures implemented include using MongoDB Cloud Manager for automated backups, logging, monitoring, and access control, as well as enabling MFA codes to enhance authentication. To prevent injection attacks in the Spring Boot microservice, libraries such as antisamy, hibernate-validator, and commons-lang3 were included in the project's pom.xml file. These libraries ensure input validation and sanitization, mitigating the risk of injection vulnerabilities. The spring-cloud-starter-gateway and spring-boot-starter-cache libraries were implemented to prevent the Spring Boot microservice against DoS attacks, further enhancing its security.

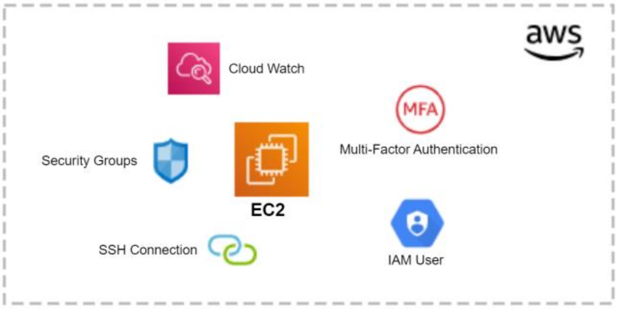


Fig. 9. EC2 Threat Mitigation

IV. Discussion

In the face of rapidly evolving technology and the widespread adoption of cloud-based environments, relying solely on a group of experts convening in a room for a limited time frame is inadequate to effectively preventing the threats targeting multiple systems. Traditional threat modeling sessions typically commence with an interactive discussion involving security experts and stakeholders [2]. These sessions often resemble a collaborative whiteboarding session, where participants engage in brainstorming and deliberation to identify risk factors and generate ideas to mitigate them [2]. Creating a threat model using ChatGPT relies on an artificial intelligence language model that can analyse vast amounts of data and generate insights based on that analysis [19]. Conventional methodologies may be limited in their ability to identify complex or emerging threats, whereas ChatGPT can potentially identify new or emerging threats by analysing large amounts of data from various sources. ChatGPT can analyse data in real-time, providing quick insights and recommendations for improving the security posture of a system. It also eliminates the need for extensive manual processes, which can save time and resources.

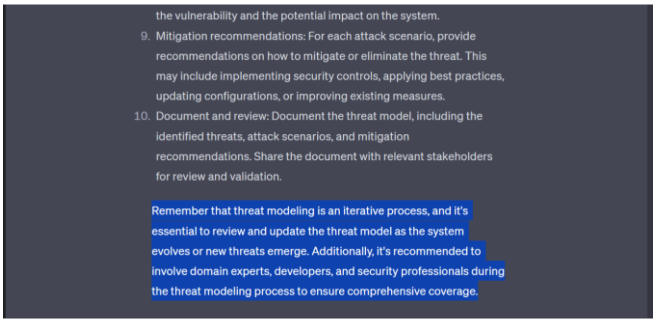
This research highlighted the efficiency of ChatGPT in identifying potential threats, particularly new and emerging ones, along with their corresponding mitigation measures for our cloud-based microservice system. A list of potential threats and corresponding mitigation measures were effortlessly provided by ChatGPT within a short span of time. The successful implementation of mitigation measures demonstrated a proactive and multi-layered approach to securing the whole system which included the AWS infrastructure, EC2 instance, Spring Boot microservice, and MongoDB database. By addressing various security aspects such as access control, network security, input validation, and DoS protection, the system is better equipped to withstand potential threats and maintain the confidentiality, integrity, and availability of the resources. Continued monitoring, regular updates, and adherence to security best practices will be crucial to ensuring the long-term security of the system and safeguarding against emerging threats. Without the assistance of ChatGPT, it would have been challenging for just human expertise to identify these threats with such speed and accuracy. The ability of ChatGPT to generate insights with just a prompt and click has been invaluable in this regard.

However, there are certain limitation for capabilities of ChatGPT. It often struggles to comprehend questions that are phrased in a specific manner, requiring rephrasing for better understanding. While initially responses may appear reasonable, they can sometimes lack practicality or become overly verbose. Therefore, we need to exercise caution when formulating prompts and interpreting the generated responses. It is crucial to double-check whether the provided information accurately meets our requirements. Since ChatGPT relies on a vast array of sources, there is a possibility of inaccurate or confusing information being presented. In order to construct the desired threat model for this paper, the prompt had to be modified with specific and additional information as per the requirements, generating multiple responses until we obtained the specific outcome we were seeking.

Though ChatGPT can suggest measures and provide ideas on threats and mitigations, it does not have the capability to practically carry out these measures. The implementation of the suggested measures requires the expertise and knowledge of humans in relevant domains. In the process of conducting this paper, it was necessary to rely on domain knowledge and established practices for implementing the measures. ChatGPT served as a valuable tool for generating ideas and insights, but it does not replace the need for human expertise in executing these measures.

While ChatGPT can provide valuable assistance, we should be cautious about becoming overly reliant on it. It is essential to maintain confidence in our own cognitive abilities, human intelligence, and diligent work ethics. We must remember that no AI or chatbot can surpass the capabilities of human intelligence. Moreover, we should not forget the fact that ChatGPT being an AI model, inherently carries numerous risk factors associated with the use of artificial intelligence. The European Parliament has recently given its approval to the EU's AI act, which adopts a risk-based strategy for governing artificial intelligence. The regulations outline specific criteria for providers of "foundation models" like ChatGPT, which have raised significant concerns among regulators due to their increasingly advanced capabilities and the potential displacement of even skilled workers [28].

Indeed, ChatGPT itself has acknowledged these limitations during the course of this research, as evident in the responses provided (Fig. 10, 11). This further strengthens the conclusion that careful consideration is necessary when relying on ChatGPT, as it highlights the challenges in comprehending specific questions and the potential for responses to lack practicality. The recognition of these limitations by ChatGPT itself supports the need for caution and thorough evaluation when utilizing its capabilities.

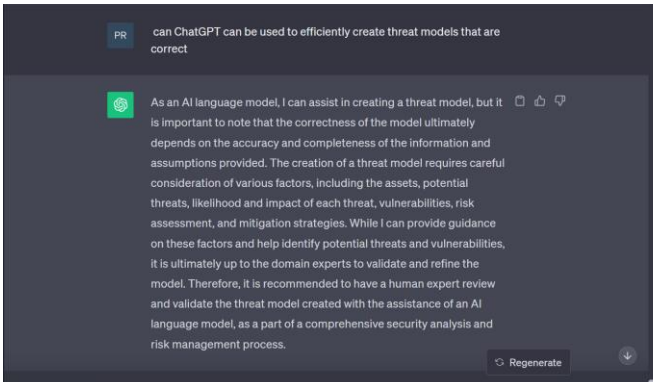


the vulnerability and the potential impact on the system.

9. Mitigation recommendations: For each attack scenario, provide recommendations on how to mitigate or eliminate the threat. This may include implementing security controls, applying best practices, updating configurations, or improving existing measures.
10. Document and review: Document the threat model, including the identified threats, attack scenarios, and mitigation recommendations. Share the document with relevant stakeholders for review and validation.

Remember that threat modeling is an iterative process, and it's essential to review and update the threat model as the system evolves or new threats emerge. Additionally, it's recommended to involve domain experts, developers, and security professionals during the threat modeling process to ensure comprehensive coverage.

Fig. 10. ChatGPT stating its limitation - A



PR can ChatGPT can be used to efficiently create threat models that are correct

As an AI language model, I can assist in creating a threat model, but it is important to note that the correctness of the model ultimately depends on the accuracy and completeness of the information and assumptions provided. The creation of a threat model requires careful consideration of various factors, including the assets, potential threats, likelihood and impact of each threat, vulnerabilities, risk assessment, and mitigation strategies. While I can provide guidance on these factors and help identify potential threats and vulnerabilities, it is ultimately up to the domain experts to validate and refine the model. Therefore, it is recommended to have a human expert review and validate the threat model created with the assistance of an AI language model, as a part of a comprehensive security analysis and risk management process.

Fig. 11. ChatGPT stating its limitation - B

To ensure the creation of a robust threat model that meets specific requirements, it is recommended to supplement ChatGPT responses with additional resources such as industry guidelines, regulations and consultations with security experts. ChatGPT can definitely provide a foundational understanding, but a comprehensive and thorough approach involving extensive research and expert input is essential for developing an effective threat model.

Acknowledgment

I express my gratitude to Ruth Lennon and Danny McFadden for their valuable guidance throughout the preparation of this research paper.

References

[1] L. Dupont, "Threat Modeling Insider -2023 Threat Modeling Connect Hackaton," Toreon - Business driven cyber consulting, Mar. 30, 2023. https://www.toreon.com/tmi-newsletter-24-a-deep-dive-into-the-2023-threat-modeling-connect-hackathon/ (accessed Jun. 01, 2023).

[2] H. N. Security, “Threat modeling needs a reset,” Help Net Security, Jun. 30, 2021. https://www.helpnetsecurity.com/2021/06/30/threat-modeling-process/ (accessed Jun. 01, 2023).

[3] T. Ijlal, “ChatGPT can boost your Threat Modeling skills,” Medium, Feb. 11, 2023. https://infosecwriteups.com/chatgpt-can-boost-your-threat-modeling-skills-ab82149d0140 (accessed May 26, 2023).

[4] K. Bernsmed, D. S. Cruzes, M. G. Jaatun, and M. Iovan, “Adopting threat modelling in agile software development projects,” J. Syst. Softw., vol. 183, p. 111090, Jan. 2022, doi: 10.1016/j.jss.2021.111090.

[5] F. Swiderski and W. Snyder, Threat Modelling. [Online]. Available: http://refhub.elsevier.com/S0164-1212(21)00187-4/sb32

[6] M. Howard and S. Lipner, The Security Development Lifecycle. [Online]. Available: http://refhub.elsevier.com/S0164-1212(21)00187-4/sb13

[7] “Threat Modeling - What, Why & How (eBook).pdf.”

[8] A. J. Lee, G. A. Koenig, and W. Yurcik, “Cluster Security with NVisionCC: The Forseti Distributed File Integrity Checker”.

[9] A. Shostack, Threat modeling: Designing for security. [Online]. Available: http://refhub.elsevier.com/S0164-1212(21)00187-4/sb28

[10] “Methodologies Blog,” Jan. 23, 2023. https://www.iriusrisk.com/resources-blog/methodologies (accessed May 10, 2023).

[11] M. T, “Top 5 Threat Modeling Methodologies,” Practical DevSecOps, Feb. 06, 2023. https://www.practical-devsecops.com/threat-modeling-methodologies/ (accessed May 10, 2023).

[12] “Threat Modeling Process and Methodologies,” Simplilearn.com, Feb. 21, 2020. https://www.simplilearn.com/what-is-threat-modeling-article (accessed Jun. 02, 2023).

[13] “Microsoft Security Development Lifecycle Threat Modelling.” https://www.microsoft.com/en-us/securityengineering/sdl/threatmodeling (accessed May 11, 2023).

[14] “OWASP Threat Dragon.” https://www.threatdragon.com/#/ (accessed May 11, 2023).

[15] E. Bygdas, L. A. Jaatun, S. B. Antonsen, A. Ringen, and E. Eiring, “Evaluating Threat Modeling Tools: Microsoft TMT versus OWASP Threat Dragon,” in 2021 International Conference on Cyber Situational Awareness, Data Analytics and Assessment (CyberSA), Dublin, Ireland: IEEE, Jun. 2021, pp. 1–7. doi: 10.1109/CyberSA52016.2021.9478215.

[16] L. Sion, D. Van Landuyt, K. Yskout, and W. Joosen, “SPARTA: Security & Privacy Architecture Through Risk-Driven Threat Assessment,” in 2018 IEEE International Conference on Software Architecture Companion (ICSA-C), Seattle, WA: IEEE, Apr. 2018, pp. 89–92. doi: 10.1109/ICSA-C.2018.00032.

[17] “What Is Threat Modeling and How Does It Work? | Synopsys.” https://www.synopsys.com/glossary/what-is-threat-modeling.html (accessed May 10, 2023).

[18] “Threat Modeling Process | OWASP Foundation.” https://owasp.org/www-community/Threat_Modeling_Process (accessed May 10, 2023).

[19] “What is ChatGPT and why does it matter? Here’s what you need to know,” ZDNET. https://www.zdnet.com/article/what-is-chatgpt-and-why-does-it-matter-heres-everything-you-need-to-know/ (accessed Jun. 02, 2023).

[20] “Introducing ChatGPT.” https://openai.com/blog/chatgpt (accessed Jun. 02, 2023).

[21] “ChatGPT.” https://chat.openai.com (accessed May 26, 2023).

[22] “Let’s chat about ChatGPT,” global. https://www.ubs.com/global/en/wealth-management/our-approach/marketnews/article.1585717.html (accessed Jun. 02, 2023).

[23] “ChatGPT - Creating Threat Model.” https://chat.openai.com (accessed May. 02, 2023).

[24] “ChatGPT - Ensuring Security of EC2 Instance and MongoDB.” https://chat.openai.com (accessed May. 2, 2023).

[25] “ChatGPT - Input Validation and Sanitization to prevent injection attacks.” https://chat.openai.com (accessed May. 2, 2023).

[26] “ChatGPT - Implement proper authentication and authorization mechanisms.” https://chat.openai.com (accessed May. 2, 2023).

[27] “ChatGPT - Implement DoS protection mechanisms.” https://chat.openai.com (accessed May. 2, 2023).

[28] R. Browne, “Europe takes aim at ChatGPT with what might soon be the West’s first A.I. law. Here’s what it means,” CNBC, May 15, 2023. https://www.cnbc.com/2023/05/15/eu-ai-act-europe-takes-aim-at-chatgpt-with-landmark-regulation.html (accessed Jun. 22, 2023).