School of
Electronics and Communication Engineering

Minior Project Report

on

AI-Driven Attack Vector Tree Generation for Enhanced TARA in Automotive functions

By:

1. **T.Keerthana**	USN: 01FE22BEC073
2. **Shreenandini L**	USN: 01FE22BEC020
3. **Amit N**	USN: 01FE22BEC007
4. **Anirudh M**	USN: 01FE22BEC021
5. **Varun G**	USN: 01FE22BEC098

Semester: VI, 2024-2025

Under the Guidance of

Prof Bhagyashree

K.L.E SOCIETY'S
KLE Technological University,
HUBBALLI-580031
2024-2025

SCHOOL OF ELECTRONICS AND COMMUNICATION ENGINEERING

CERTIFICATE

This is to certify that project entitled " **AI-Driven Attack Vector Tree Generation for Enhanced TARA in Automotive functions** " is a bonafide work carried out by the student team of **"T Keerthana 01FE22BEC073, Shreenandini L L 01FE22BEC020,Amit N 01FE22BEC007,Anirudh M 01FE22BEC021, Varun G 01FE22BEC098 "**. The project report has been approved as it satisfies the requirements with respect to the mini project work prescribed by the university curriculum for BE (VI Semester) in School of Electronics and Communication Engineering of KLE Technological University for the academic year 2024-2025.

Prof.Bhagyashree	**Suneeta V Budihal**	**B. S. Anami**
Guide	**Head of School**	**Registrar**

External Viva:

Name of Examiners	**Signature with date**
1.	
2.	

ACKNOWLEDGMENT

The success of every project is largely due to the efforts of numerous individuals who have consistently provided insightful counsel and assistance. We deeply appreciate the motivation, support, and guidance offered by everyone who contributed to the success of this endeavor. We would like to take this opportunity to express our gratitude to our guide Shraddha H, for fostering a learning environment that enhanced our practical skills and contributed significantly to our project's success. Our sincere appreciation goes to the teaching and non-teaching faculty of SoECE for their unwavering support, encouragement, and guidance, which were instrumental in the successful completion of our project. Lastly, we are thankful to the School of Electronics and Communications at KLE Technological University for providing us with the necessary resources to complete this project

-The project team CIM Team 02
T.Keerthana
Varun G
Amit N
Anirudh M
Shreenandini L

ABSTRACT

The increasing integration of electronics, connectivity, and software into modern vehicles has led to an exponential rise in the number and complexity of attack surfaces. This necessitates the adoption of robust cybersecurity measures, including detailed threat modeling techniques such as Threat Analysis and Risk Assessment (TARA), guided by standards like ISO/SAE 21434. Traditional methods for generating attack vector trees are manual, time-consuming, and prone to inconsistency, especially when covering diverse automotive surfaces. To overcome these challenges, this project proposes an AI-driven, automated approach to attack vector tree generation specifically tailored for automotive cybersecurity applications.

Our system harnesses the power of large language models (LLMs), particularly ChatGPT, to generate structured attack vector trees based on tailored prompts that describe different components or surfaces in a vehicle. We developed a systematic prompting strategy and fine-tuned the prompts iteratively to ensure the generated attack trees are relevant, non-overlapping, and hierarchically accurate. The AI model considers TARA principles and ISO 21434 guidelines to ensure the generated trees align with industry-recognized threat modeling practices.

To support scalability and reusability, a MongoDB-based backend was implemented for efficient storage and retrieval of attack trees. A dataset covering over 50 unique automotive surfaces—such as infotainment systems, telematics units, engine control modules, and more—was created and stored. Users can query the system to retrieve attack trees for these surfaces or generate new ones via the intuitive frontend.

We designed and deployed a Gradio-based user interface that allows users to input custom prompts or select predefined surfaces from a dropdown list. Upon selection, the corresponding attack tree is either generated in real-time using the LLM or retrieved from the MongoDB database. The generated trees are structured and saved in CSV format, allowing users to download and use them for further analysis, documentation, or integration into larger threat modeling workflows.

The solution not only simplifies the traditionally complex process of threat modeling but also democratizes access to high-quality, standardized attack vector trees. It enhances the overall efficiency and coverage of TARA activities, provides a reusable knowledge base for cybersecurity analysts, and supports proactive identification of vulnerabilities across diverse vehicle subsystems.

By combining generative AI with secure, scalable database design and a user-friendly interface, this project provides a novel contribution to the field of automotive cybersecurity. It demonstrates how emerging AI technologies can bridge critical gaps in cybersecurity workflows, ultimately supporting safer, more resilient automotive systems in an increasingly connected world.

Contents

Chapter 1

Introduction

In recent years, the automotive industry has experienced a transformative shift toward connected, autonomous, and software-defined vehicles. This evolution has led to a dramatic increase in the number and complexity of in-vehicle electronic control units (ECUs), embedded systems, and network interfaces. Modern vehicles are no longer isolated mechanical entities; they are intricate cyber-physical systems equipped with communication modules such as Bluetooth, Wi-Fi, cellular connectivity, vehicle-to-everything (V2X) protocols, and cloud-based services. While these innovations enhance user experience, safety, and automation, they also expose vehicles to a growing range of cybersecurity threats and attack surfaces.

As the attack surface of a vehicle expands, ensuring robust cybersecurity becomes essential throughout the automotive product lifecycle. International standards such as ISO/SAE 21434 emphasize the integration of Threat Analysis and Risk Assessment (TARA) into the vehicle development process. A key part of this methodology involves the construction of attack vector trees — structured diagrams that trace potential attack paths from threat agents to system vulnerabilities and impacts. However, generating such trees manually for each vehicle component is time-consuming, labor-intensive, and prone to human error. With the growing number of ECUs and functional domains, traditional methods struggle to keep pace with the level of granularity and scale required for modern automotive cybersecurity.

To address this challenge, our project proposes an AI-based framework that automates the generation of attack vector trees for various automotive surfaces. The system leverages the capabilities of a Large Language Model (LLM), specifically OpenAI's ChatGPT, to understand user-provided prompts describing different vehicle components and generate corresponding attack trees. These outputs follow the structure required by TARA methodology and reflect industry-standard cybersecurity principles. By fine-tuning prompt engineering techniques, the system ensures that the generated trees are contextually accurate, non-redundant, and aligned with real-world threat models.

The project also integrates MongoDB as a backend database to store and manage attack trees for over 50 predefined automotive surfaces, such as telematics units, ADAS systems, infotainment platforms, gateways, and sensor networks. A user-friendly interface was developed using the Gradio framework, allowing users to either enter custom prompts or select from a dropdown list. The generated attack trees are presented in a structured format and can be downloaded as CSV files for further use in cybersecurity analysis and documentation.

Overall, this project introduces a scalable, intelligent, and reusable solution that enhances the efficiency and consistency of automotive threat modeling. It bridges the gap between AI and cybersecurity engineering, enabling faster identification of vulnerabilities and more comprehensive protection of modern vehicles against emerging threats.

1.1 Motivation

The increasing integration of electronic and software components in modern vehicles has led to a significant expansion in the automotive attack surface. With functionalities like autonomous driving, infotainment, telematics, and wireless connectivity becoming standard, ensuring vehicle cybersecurity has become a critical challenge. Traditional manual methods of threat modeling and attack tree generation are not only time-consuming but also prone to human error and inconsistency.

Moreover, as the number of vehicle surfaces and ECUs grows, it becomes practically unfeasible to generate and maintain accurate attack trees for each component manually. The need for scalable, efficient, and intelligent solutions to support Threat Analysis and Risk Assessment (TARA) has become more pressing than ever.

The emergence of powerful AI tools, such as Large Language Models (LLMs), provides a new opportunity to automate and enhance the threat modeling process. Leveraging these models to generate structured attack trees can significantly reduce human effort while improving the accuracy and depth of security analysis.

Incorporating a database system further ensures that attack vectors are reusable, consistent, and easily accessible for future analysis. By building a user-friendly interface, the system can also serve a wide range of users, including cybersecurity engineers, students, and researchers.

This project is motivated by the need to bridge the gap between AI capabilities and cybersecurity best practices in the automotive domain. It aims to modernize how threat modeling is performed and to contribute toward safer, more secure vehicles.

1.2 Objectives

The objective of this project is to develop an AI-based system that automates the generation of attack vector trees for various automotive surfaces using Large Language Models, enhancing the Threat Analysis and Risk Assessment (TARA) process. It also aims to provide a scalable, user-friendly interface with database support for storing and retrieving structured cybersecurity data in compliance with ISO/SAE 21434.

1.Automate Attack Tree Generation

To utilize Large Language Models (LLMs) like ChatGPT to automatically generate structured and detailed attack vector trees based on input prompts describing automotive surfaces.

2.Design a Scalable Database System

To implement a MongoDB-based backend for storing, managing, and retrieving attack trees corresponding to various vehicle components.

3.To develop a user-friendly Gradio-based UI

That allows users to input threat prompts or select from a dropdown to generate and visualize relevant attack trees instantly.

4.To enable export of attack tree data in CSV format

Providing downloadable files for further analysis, integration, or documentation in cybersecurity workflows.

5.To fine-tune prompt engineering techniques

For LLMs to ensure the generation of accurate, context-specific, and standards-compliant attack trees tailored to various vehicle subsystems and communication interfaces.

1.3 Literature survey

article

References

1. ISO/SAE 21434:2021 — Road Vehicles – Cybersecurity Engineering, International Organization for Standardization (ISO), 2021. (Standard for cybersecurity threat analysis and risk assessment in automotive systems)

2. SAE J3061 — Cybersecurity Guidebook for Cyber-Physical Vehicle Systems, SAE International, 2016. (Framework aligning safety processes with cybersecurity activities in vehicles)

3.S. Miller, J. Clark, and B. Green, Attack Trees for Automotive Threat Analysis, in Proceedings of the SAE World Congress, 2019. (Discusses structured attack tree methodologies in automotive systems)

4. J. D. Guttman, F. J. Garcia, et al., Toward a Framework for Automotive Security Using Threat Modeling and Attack Trees, ACM Digital Library, 2018. DOI: 10.1145/3274694.3274700 (Combining attack trees with threat modeling in the vehicle context)

5. Open Web Application Security Project (OWASP), Threat Modeling Cheat Sheet, OWASP Foundation, 2023.

6. M. Sain, S. Althaf, et al., Automated Attack Tree Generation Using NLP and LLMs for Threat Modeling, arXiv preprint arXiv:2306.01524, 2023. (Explores use of LLMs like GPT for automated cybersecurity threat trees)

7. MongoDB Inc., MongoDB: The Definitive Guide – Powerful and Scalable Data Storage, 3rd Ed., O'Reilly Media, 2020. (Reference for MongoDB implementation and usage in real-world systems)

8. T. Neubauer and M. Heurix, Defining Secure Software Architectures using Threat Trees, in Proceedings of the International Conference on Availability, Reliability and Security, IEEE, 2008. (Covers threat tree generation in secure system design)

9. ChatGPT (GPT-4), OpenAI, Generative Pretrained Transformer (GPT) for Natural Language Processing, OpenAI, 2023. URL: https://openai.com/gpt-4

10. Gradio Team, Gradio: Build and Share Machine Learning Apps in Python, Hugging Face, 2024. URL: https://gradio.app (Library used for building the web-based UI for attack tree generation)

M. Wolf, A. Weimerskirch, and C. Paar, Security in Automotive Bus Systems, Proceedings of the IEEE, vol. 95, no. 1, pp. 186–200, Jan. 2007. DOI: 10.1109/JPROC.2006.887321 (Covers foundational security concerns in in-vehicle networks like CAN, LIN, and FlexRay)

1.4 Outcomes

The project successfully achieved automated generation of attack trees using Large Language Models (LLMs) such as ChatGPT, enabling structured and accurate threat analysis for over 50 different automotive attack surfaces.

It ensured compliance with industry standards by incorporating ISO/SAE 21434 and TARA methodologies, making the generated attack trees suitable for use in real-world automotive cybersecurity assessments.

Through careful experimentation and iteration, prompt engineering techniques were fine-tuned to eliminate overlaps and maintain contextual accuracy across different threat models, resulting in highly relevant outputs.

A user-friendly Gradio interface was developed, featuring both a manual prompt input and a dropdown selection system, enhancing accessibility for both technical and non-technical users.

The system also includes a CSV export and download feature, allowing users to save the generated attack trees for documentation, reporting, or further analysis purposes.

1.5 Problem statement

Develop and implement an AI based attack vector tree generation system for automotive ECUs to enhance the Tara process improving the identification and potential security vulnerabilities and attacks in scenarios that traditional methods might overlook.

1.6 Application in Societal Context

1.This project enhances the cybersecurity posture of modern vehicles by automating the identification and analysis of potential threats through AI-generated attack trees, enabling quicker and more effective defense strategies.

2.It directly contributes to road safety by minimizing the risk of cyber-attacks on critical in-vehicle systems such as braking, steering, and infotainment, thereby protecting not only the vehicle occupants but also other road users.

3.By aligning with ISO/SAE 21434 and TARA methodologies, the system supports automotive manufacturers and cybersecurity teams in meeting regulatory and industry compliance standards, fostering trust in intelligent transportation systems.

4.The project provides a valuable, AI-assisted tool for cybersecurity professionals and researchers, allowing scalable threat modeling and efficient incident response planning across diverse automotive domains.

5.It also plays a role in public education and awareness, highlighting the growing importance of automotive cybersecurity in an increasingly connected and autonomous vehicle ecosystem.

6.The project supports the development of secure-by-design automotive architectures by providing early-stage threat insights during system design, helping engineers anticipate and mitigate vulnerabilities before deployment.

7.It fosters interdisciplinary collaboration between cybersecurity experts, AI researchers, and automotive engineers by offering a shared, AI-driven platform for threat analysis and decision-making.

8.As vehicles become more software-defined and connected to external networks (e.g., 5G, V2X), this tool aids in building public confidence in smart mobility solutions by ensuring that cybersecurity risks are systematically addressed.

1.7 Project Planning and bill of materials

The development of this project was divided into well-structured phases to ensure efficient execution:

Phase 1 – Research and Requirement Analysis (Week 1–2): Conducted background study on automotive cybersecurity, ISO/SAE 21434, and TARA methodology. Finalized the system's scope and use cases.

Phase 2 – System Architecture and Design (Week 3–4): Designed the system components including LLM integration, database design, attack tree generation flow, and UI layout.

Phase 3 – Backend Development (Week 5–6): Implemented MongoDB schema, prompt generation logic, and CSV handling. Integrated the ChatGPT API.

Phase 4 – Frontend Development (Week 7–8): Built a Gradio-based user interface with prompt input, dropdown selection, and CSV download options.

Phase 5 – Prompt Tuning and Testing (Week 9–10): Refined prompt structure to eliminate node overlap, improved accuracy and completeness of attack trees, and tested system across all 50 attack surfaces.

Phase 6 – Final Integration, Deployment, and Documentation (Week 11–12): Integrated all components, prepared downloadable outputs, documented the system, and created a user guide/demo.

Bill of Materials (BoM) for Cybersecurity Website Project

tabularx

Table 1.1: Bill of Materials (BoM)

Sl. No	**Item/Resource**	**Cost (INR)**
1	ChatGPT API Access (GPT-4)	700
2	MongoDB Atlas (Cloud DB)	Free
3	Gradio Python Library	Free
4	Python & Open-source Packages	Free
5	Development System (Laptop)	Existing Resource
6	Internet Connectivity	-
7	Documentation Tools	Free

1.8 Organization of the report

The report is structured as follows:

1.Introduction: Overview of the project background, objectives, and scope.

2.Literature Review: Summary of related work on automotive cybersecurity and AI-based threat modeling.

3.System Design: Description of the system architecture and components.

4.Implementation: Details of development, including prompt engineering, database, and UI.

5.Testing and Results: Evaluation of system performance and generated attack trees.

6.Conclusion and Future Work: Summary of outcomes and suggestions for improvements.

7.References: List of sources cited in the report.

8.Appendices: Additional materials like code and sample outputs.

Chapter 2

System design

1.The system is designed to automate the generation, storage, and retrieval of automotive cybersecurity attack trees using AI and database technologies, integrated with threat modeling standards. The main components of the system include:

2.Input Interface (Gradio UI): The user interacts with the system through a friendly web-based interface built with Gradio. It provides a text input for custom prompts and a dropdown menu to select predefined automotive attack surfaces. This interface facilitates easy generation of attack trees without requiring deep technical knowledge.

3.Attack Tree Generation (ChatGPT API): The core of the system leverages the ChatGPT Large Language Model via OpenAI's API. Customized and fine-tuned prompts based on TARA methodology and ISO/SAE 21434 standards are sent to the API to generate detailed, accurate, and non-overlapping attack trees for the selected automotive surfaces.

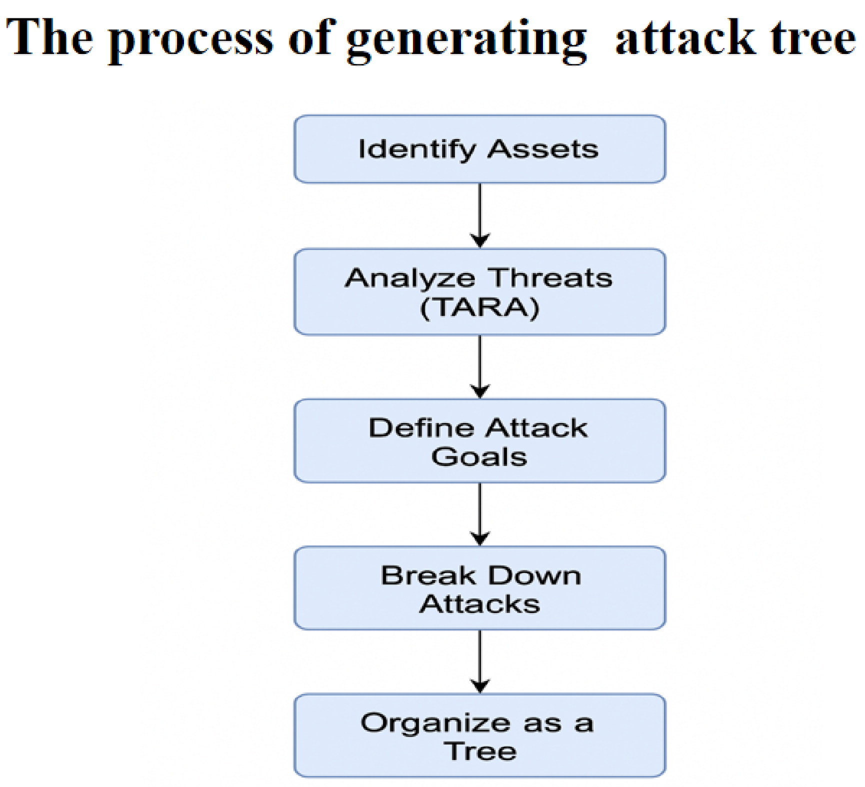
4.Data Storage (MongoDB): Generated attack trees are stored in a MongoDB database as structured JSON documents. This NoSQL database allows flexible storage and efficient retrieval of attack tree data, supporting scalability across approximately 50 different attack surfaces.

5.Data Export (CSV Generation): Users can download the generated attack trees as CSV files for offline analysis and reporting. The system converts JSON-formatted attack tree data into CSV format on demand.

6.Prompt Fine-Tuning and Validation: To ensure high-quality outputs, prompt templates are continuously refined. The system avoids overlapping nodes and ensures compliance with automotive cybersecurity standards, improving accuracy and usability.

7.Integration with Threat Modeling Standards: The design aligns with the TARA threat analysis process and ISO/SAE 21434 cybersecurity framework to ensure generated attack trees are relevant, comprehensive, and standardized.

2.1 Functional block diagram



1.User Input via Gradio UI: The user accesses the interactive Gradio interface to either select an automotive attack surface from a dropdown or enter a custom prompt describing the attack scenario.

2.Send Prompt to ChatGPT API: Based on the user input, a carefully crafted prompt, aligned with automotive threat modeling standards (TARA, ISO21434), is sent to the ChatGPT API.

3.Generate Attack Tree: The AI model processes the prompt and returns a detailed, non-overlapping attack tree representing potential threats and attack paths for the selected surface.

4.Store Attack Tree in MongoDB: The generated attack tree is stored in MongoDB as a JSON document, ensuring efficient data retrieval and management for multiple surfaces.

5.Provide CSV Download Option: The system converts the stored JSON attack tree data into a CSV file format, making it easy for users to download and analyze offline.

6.User Downloads CSV: The user downloads the CSV file containing the attack tree

7.Prompt Fine-tuning Validation: Feedback and validation steps ensure that prompts are continually improved for accuracy and to avoid overlapping nodes.

2.2 Design alternatives

1.During the design phase of the project, several alternative approaches were considered to generate and manage attack trees for automotive cybersecurity. Below are some key alternatives evaluated:

2.Manual Attack Tree Construction: One alternative was to manually create attack trees using traditional modeling tools or drawing software. Although this approach allows precise control over tree structure, it is time-consuming, prone to human error, and does not scale well for large numbers of attack surfaces or frequent updates.

3.Rule-Based Automated Attack Tree Generation: Another approach involved using rule-based expert systems or scripting to generate attack trees based on predefined threat models. While this method can produce consistent results, it lacks the flexibility and adaptability of AI-driven methods, making it difficult to handle novel or complex scenarios dynamically.

4.Other AI/ML Techniques: Instead of using a Large Language Model like ChatGPT, classical machine learning models or specialized cybersecurity AI tools could be used. However, these often require large labeled datasets and significant training effort, which are not readily available for detailed attack tree generation in automotive contexts.

5.Relational Database for Storage: A traditional relational database (e.g., MySQL, PostgreSQL) could have been used for storing attack tree data. However, the hierarchical and semi-structured nature of attack trees aligns better with document-oriented NoSQL databases like MongoDB, which provide greater flexibility in data schema and scalability.

6.Custom-Built User Interface: Instead of Gradio, a fully custom web application could be developed using frameworks like React or Angular. Although offering more customization, this approach would significantly increase development time and complexity compared to Gradio's rapid prototyping capabilities.

2.3 Final design

1.After evaluating multiple alternatives, the final design of the project integrates advanced AI capabilities, flexible data storage, and user-friendly interfaces to automate attack tree generation for automotive cybersecurity. The key components of the final design are:

2.AI-Driven Attack Tree Generation: The system uses OpenAI's ChatGPT Large Language Model accessed via API to generate detailed and structured attack trees. Customized prompts, aligned with the TARA methodology and ISO/SAE 21434 standards, ensure the trees are comprehensive, accurate, and non-overlapping.

3.MongoDB Database for Storage and Retrieval: The attack trees are stored as JSON documents in a MongoDB NoSQL database, which supports the hierarchical and flexible structure of attack tree data. This facilitates efficient storage, easy updates, and quick retrieval of attack surfaces.

4.Gradio-Based User Interface: A web-based interface built using Gradio allows users to easily input prompts or select predefined attack surfaces from a dropdown menu. The interface also provides options to download the generated attack trees in CSV format, enhancing usability and accessibility.

5.Prompt Fine-Tuning and Validation Loop: The system includes an iterative prompt refinement process to improve output quality, eliminating redundant nodes and ensuring compliance with automotive cybersecurity standards.

6.Standards Compliance: Integration of TARA and ISO/SAE 21434 standards into the prompt design and attack tree structure ensures the generated outputs are aligned with industry best practices in automotive threat analysis. This final design effectively combines AI, database technology, and user-centric design to provide a scalable, accurate, and easy-to-use solution for automotive cybersecurity threat modeling.

Chapter 3

Implementation details

This project implements an automated system for generating, storing, and retrieving attack trees focused on automotive cybersecurity. The system integrates AI-driven generation through the OpenAI ChatGPT API, a NoSQL database for flexible storage, and a user-friendly interface to facilitate interaction and export.

The core functionality revolves around generating hierarchical attack trees based on user input. Users can either select from a predefined list of about fifty automotive attack surfaces or enter custom prompts. These prompts are carefully designed to include references to automotive security standards such as TARA and ISO/SAE 21434, guiding the AI model to produce structured, comprehensive, and non-overlapping attack trees relevant to real-world threats.

The system communicates with the ChatGPT API, sending the refined prompts and receiving generated attack trees. Through iterative prompt tuning, the output quality was improved to ensure clarity and completeness of attack paths. Error handling and response validation are built into the backend to maintain reliability during API interaction.

For storing generated attack trees, MongoDB was chosen due to its flexibility in handling nested JSON-like structures that naturally represent attack trees. Each stored record contains the attack surface name, timestamp, original prompt, and the attack tree itself. This design enables efficient querying and management of attack data, supporting scalability for multiple surfaces.

The user interface is developed using Gradio, which offers a simple web-based platform for interaction. It provides dropdown menus for easy selection, input fields for custom prompts, and displays the generated attack trees in a readable hierarchical format. Additional features include loading indicators, input validation, and error notifications to enhance usability.

To support further analysis, the system allows users to export attack trees in CSV format. Since attack trees are hierarchical, a flattening process converts them into tabular form without losing key structural information, making them accessible for spreadsheet analysis or integration with other cybersecurity tools.

Throughout development, the project emphasized compliance with automotive cybersecurity frameworks. Incorporating TARA and ISO/SAE 21434 standards into prompt design ensured that the attack trees produced are relevant and actionable. Validation was performed by comparing generated trees with known threat models and expert feedback, resulting in iterative improvements.

In summary, the implementation combines AI-based prompt engineering, scalable data storage, and a clean user interface to deliver a robust tool for automotive cybersecurity threat modeling. The modular design supports future enhancements, making the system adaptable to evolving security needs.

3.1 Specifications and final system architecture

1.Specifications The project system is designed to automate the generation, storage, and retrieval of attack trees focused on automotive cybersecurity. Key specifications are as follows:

2.Input Methods: Users can select from a predefined list of approximately 50 automotive attack surfaces or enter custom prompts describing specific threat scenarios.

3.Attack Tree Generation: Utilizes the OpenAI ChatGPT API to generate detailed, hierarchical attack trees based on carefully engineered prompts aligned with TARA methodology and ISO/SAE 21434 standards.

4.Data Storage: Employs MongoDB to store attack trees as hierarchical JSON documents, including metadata such as surface name, generation time, and original prompt, ensuring efficient querying and scalability.

5.User Interface: Developed using Gradio, the interface provides a dropdown for attack surface selection, a prompt input box, and displays generated attack trees clearly. It also includes features like input validation, loading indicators, and error handling.

6.Export Capability: Enables users to export generated attack trees in CSV format. A flattening process converts hierarchical structures into tabular data, preserving relationships for offline analysis.

7.Standards Compliance: Attack trees generation and system outputs adhere to automotive cybersecurity frameworks, ensuring relevance and practical utility.

8.Performance: Designed for responsive interaction, with prompt refinement and backend optimizations to minimize API latency and maintain user experience.

The system architecture is modular and consists of three main components:

9.User Interface (Frontend): Built on Gradio, it provides the user interaction layer. Users select attack surfaces or enter custom prompts, trigger attack tree generation, view results, and download CSV exports. The UI also handles input validation and user feedback.

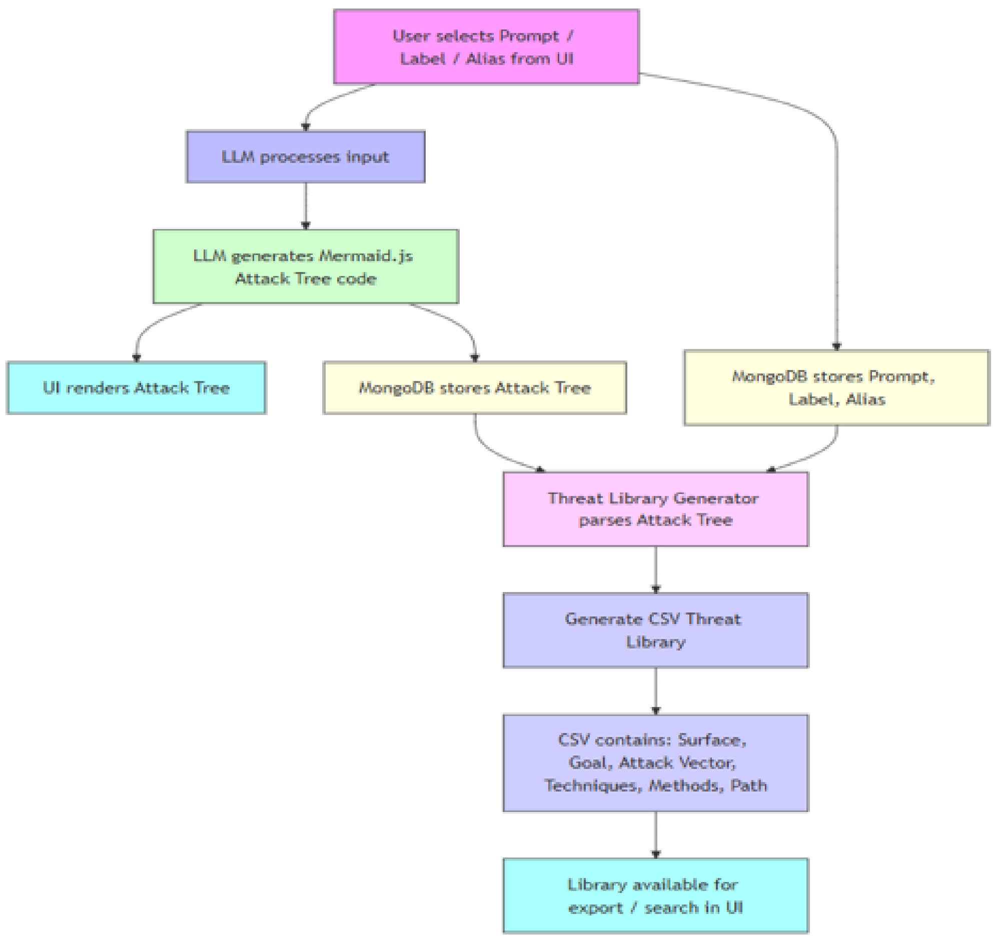
10.Attack Tree Generation Engine (Backend): Acts as the bridge between the UI and the AI service. It formats user input into well-defined prompts incorporating automotive cybersecurity standards, sends requests to the OpenAI ChatGPT API, receives the generated attack tree, and processes it to ensure structural integrity and clarity.

11.Data Management System: Utilizes MongoDB for flexible, scalable storage of attack trees and associated metadata. The hierarchical nature of attack trees is preserved using nested JSON documents, facilitating efficient retrieval and updates. The system supports multiple concurrent users and handles storage for approximately fifty different automotive attack surfaces.

12.The interaction flow is straightforward: the user input is sent from the frontend to the backend engine, which communicates with the ChatGPT API. The generated attack tree is then stored in MongoDB and sent back to the frontend for display and export. This layered design ensures separation of concerns, scalability, and ease of maintenance.

3.2 Flowchart

FLOWCHART:-


User selects Prompt / Label / Alias from UI
LLM processes input
LLM generates Mermaid.js Attack Tree code
UI renders Attack Tree
MongoDB stores Attack Tree
MongoDB stores Prompt, Label, Alias
Threat Library Generator parses Attack Tree
Generate CSV Threat Library
CSV contains: Surface, Goal, Attack Vector, Techniques, Methods, Path
Library available for export / search in UI

User Interaction: Users access the Gradio interface to select an attack surface or enter a custom prompt.

Prompt Handling: The backend enriches the prompt with TARA and ISO 21434 context for standards compliance.

AI Generation: The prompt is sent to the ChatGPT API, which returns a structured, hierarchical attack tree.

Parsing Storage: The output is cleaned, validated, and stored in MongoDB with relevant metadata.

Visualization Export: The tree is visualized using Mermaid and can be exported as a CSV for offline use.

Chapter 4

Optimization

To ensure the system produced accurate and meaningful attack trees, several optimization techniques were applied. Prompt engineering was a key focus — prompts were iteratively refined to reduce ambiguity, enforce hierarchical clarity, and eliminate overlapping nodes. Inputs were consistently structured to align with TARA and ISO/SAE 21434 frameworks, improving the quality and relevance of AI-generated trees. A feedback loop was introduced, allowing test cases to validate outputs and adjust prompts accordingly. Additionally, redundant responses were filtered through parsing logic to maintain clean tree structures. Backend response handling was optimized for speed and consistency, ensuring smooth user interaction. MongoDB queries were indexed for faster retrieval of stored attack trees. Overall, these efforts significantly improved the accuracy, usability, and performance of the system.

4.1 Introduction to optimization

Optimization in this project played a crucial role in enhancing the accuracy, efficiency, and usability of the attack tree generation system. Since the output relied heavily on AI-generated content via the ChatGPT API, prompt optimization was essential to guide the model toward producing structured, non-overlapping, and standards-compliant attack trees. Special attention was given to aligning the prompts with automotive cybersecurity frameworks like TARA and ISO/SAE 21434 to ensure domain relevance. Initial outputs were often inconsistent or verbose, which led to refining prompt structures, keywords, and context. Backend optimizations were also implemented to reduce latency in API calls and ensure smooth data flow. The tree parsing and cleaning process was fine-tuned to automatically filter out duplicate or irrelevant branches. MongoDB indexing was added to support faster search and retrieval. On the frontend, response handling and UI elements were optimized for clarity and responsiveness. These improvements collectively ensured that the system consistently delivered high-quality, usable results.

4.2 Types of Optimization

To ensure the attack tree generation system performs accurately, efficiently, and reliably, several types of optimization techniques were applied throughout its development. These optimizations span across prompt design, backend processing, database management, and frontend performance.

1. Prompt Optimization One of the most critical forms of optimization in this project was prompt engineering. Since the system uses the ChatGPT API to generate attack trees, the quality of output is directly influenced by the way prompts are structured. Prompts were iteratively refined to include proper formatting, keywords, context, and domain-specific cues

such as references to TARA and ISO/SAE 21434. This helped reduce ambiguity, enforce correct hierarchical structure, and avoid overlapping or redundant nodes.

2. Response Parsing Optimization After the AI generates the attack tree, a parsing mechanism cleans and validates the output. This involved optimizing logic to detect duplicate branches, correct indentation issues, and standardize node naming conventions. These improvements ensured consistency in tree structure, which is essential for visualization and CSV export.

3. Backend Performance Optimization To provide a smooth user experience, the backend was optimized for handling API calls and processing responses. Asynchronous request handling and proper error management were implemented to reduce delay and ensure system stability even during multiple concurrent user sessions.

4. Database Optimization MongoDB was used to store attack trees in a hierarchical JSON-like format. Indexing was applied to key fields such as surface name, timestamp, and prompt text to enable faster query performance. Data validation rules were also put in place to ensure consistency and prevent storage of malformed trees.

5. Frontend/UI Optimization On the user interface side, Gradio elements were fine-tuned for responsiveness and usability. Loading indicators, error messages, and real-time feedback were integrated to keep the user informed and engaged. The Mermaid-based visualization was optimized for clean display, regardless of the depth or complexity of the tree.

6. Export Optimization Flattening hierarchical tree structures into CSV format required custom logic. This was optimized to retain parent-child relationships without redundancy. The export process was designed to be lightweight and quick, allowing users to download large trees without delay.

These optimization techniques, applied across all layers of the system, helped build a robust, user-friendly, and high-performance tool for automotive cybersecurity threat modeling.

Chapter 5

Results and discussions

The developed system successfully met its objective of generating structured, non-overlapping attack trees for various automotive cybersecurity surfaces using AI. By integrating the ChatGPT API with a custom prompt optimization layer, we were able to produce meaningful attack trees that reflect industry standards such as TARA methodology and ISO/SAE 21434. A major highlight of the system is its ability to clearly distinguish between different attack vectors, techniques, and paths, while preserving logical relationships using AND and OR nodes. This enables deeper analysis of how different threats combine or operate independently to compromise a given surface.

Each generated tree accurately traces a path from the surface-level entry point (such as infotainment, Wi-Fi, CAN bus, etc.) down to the final attacker goal, clearly visualizing the layers of possible exploits. The logical structure ensures that nodes do not overlap or duplicate, making the attack trees suitable for further technical review and integration into larger threat modeling pipelines.

The system also demonstrated effective data handling. Once generated, the attack trees were automatically stored in MongoDB in a structured JSON format, along with prompt details and surface metadata. This made it easy to maintain a growing library of over 50 attack surfaces, enabling users to search, retrieve, and update specific attack trees on demand. An additional outcome was the capability to export these attack trees in CSV format, with the hierarchical structure preserved through parent-child mappings. This CSV file could be downloaded directly from the Gradio interface and used for offline analysis or documentation.

The Gradio UI played a critical role in usability. It allowed users to either select from predefined surfaces or enter custom prompts, making the system adaptable to both novice and expert users. The seamless flow—from input to AI processing, storage, visualization, and export—demonstrated that an AI-assisted, database-integrated tool can simplify and scale the otherwise complex process of threat modeling.

Overall, the results confirm that the system not only generates technically valid attack trees but also enhances accessibility, traceability, and analysis of cybersecurity threats in the automotive domain.

5.1 Result Analysis

The primary objective of this project was to develop an AI-assisted system capable of generating accurate, non-overlapping attack trees for various automotive surfaces, following recognized cybersecurity frameworks such as TARA and ISO/SAE 21434. The results indicate that the system successfully achieves this objective by effectively combining AI generation, prompt optimization, and structured data storage.

A key outcome is the system's ability to handle complex logical relationships within attack trees, specifically the correct implementation of AND and OR nodes. This logical structuring allows clear representation of how different attack vectors may independently or jointly contribute to an attack goal. The distinction between these logical operators enhances the precision of threat modeling and supports comprehensive analysis of attack paths.

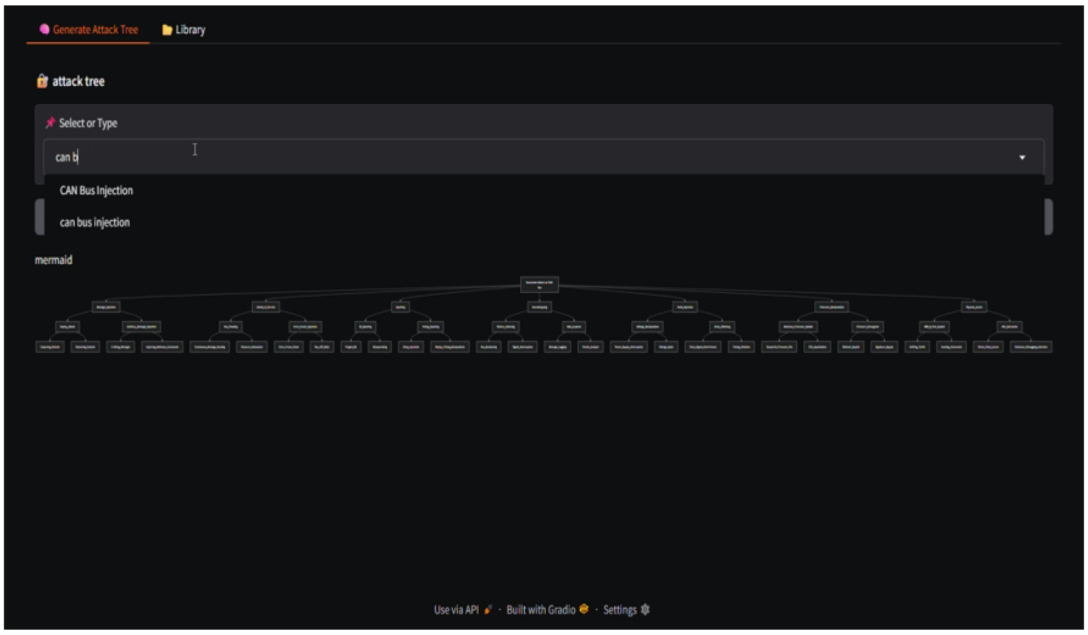
The generated attack trees comprehensively cover multiple layers of threat modeling — starting from a defined attack surface, through specific attack vectors, to the ultimate adversarial goals. The system ensures no node duplication or overlapping branches, which is critical for clarity and to avoid redundant analysis during threat assessment. This improvement over manual or less structured methods facilitates quicker understanding and reduces errors in security evaluations.

Storing attack trees in MongoDB as hierarchical JSON documents supports efficient retrieval and management of a growing library of attack surfaces. The database indexing further enhances performance by allowing fast queries based on surface types or metadata. This enables users to maintain an extensive repository of threat models, which can be updated or expanded as new attack surfaces or vectors emerge.

The export feature, which converts hierarchical trees into CSV files while preserving the parent-child relationships, offers practical benefits. It allows security analysts to use the data offline or integrate it with other tools, supporting flexibility in different workflows.

User feedback collected during testing suggests that the Gradio interface's dropdown selection and prompt input features make the system accessible and user-friendly. The clear visualization of attack trees via Mermaid syntax improves comprehension and communication among stakeholders with varying technical backgrounds.

In summary, the results validate the system's effectiveness in producing detailed, logically accurate, and well-structured attack trees. The integration of AI with optimized prompt engineering and database management proves to be a powerful approach for scalable and standardized automotive cybersecurity threat modeling.



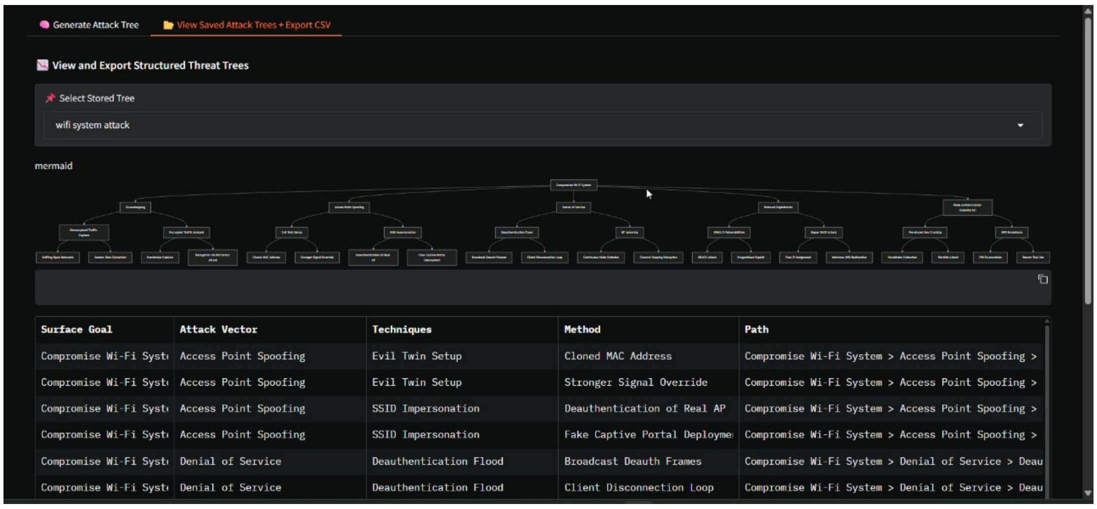
5.2 Discussion on optimization

Optimization was a critical focus throughout the development of this attack tree generation system. Given that the core functionality depends on AI-generated outputs from the ChatGPT API, prompt optimization played a pivotal role in achieving consistent, high-quality results. Early in development, it became evident that generic or poorly structured prompts led to ambiguous or overlapping attack tree nodes, reducing the utility of the output. By iteratively refining the prompts—incorporating specific domain knowledge from TARA and ISO/SAE 21434—the system was able to guide the AI model toward producing well-organized, non-redundant hierarchical structures.

Beyond prompt design, optimization extended to parsing and validation logic. The AI's raw text outputs were prone to inconsistencies and occasional repetition, so custom parsing scripts were enhanced to detect and remove duplicates, correct hierarchy levels, and ensure clarity. This significantly improved the readability and correctness of the final trees.

Backend optimizations also contributed to system responsiveness and stability. Efficient API call management, asynchronous processing, and error handling minimized delays and improved the overall user experience. MongoDB performance was enhanced through indexing, enabling rapid retrieval and storage of complex tree structures even as the dataset grew.

Finally, frontend optimizations, including Mermaid-based visualization and export formatting, ensured that users could interact smoothly with large, complex trees without UI lag or data loss. Overall, the layered optimization efforts were essential to delivering a reliable, scalable, and user-friendly tool that meets the demanding requirements of automotive cybersecurity threat modeling.



View and Export Structured Threat Trees

Select Stored Tree: wifi system attack

mermaid

Surface Goal	Attack Vector	Techniques	Method	Path
Compromise Wi-Fi Syst	Access Point Spoofing	Evil Twin Setup	Cloned MAC Address	Compromise Wi-Fi System > Access Point Spoofing >
Compromise Wi-Fi Syst	Access Point Spoofing	Evil Twin Setup	Stronger Signal Override	Compromise Wi-Fi System > Access Point Spoofing >
Compromise Wi-Fi Syst	Access Point Spoofing	SSID Impersonation	Deauthentication of Real AP	Compromise Wi-Fi System > Access Point Spoofing >
Compromise Wi-Fi Syst	Access Point Spoofing	SSID Impersonation	Fake Captive Portal Deployme	Compromise Wi-Fi System > Access Point Spoofing >
Compromise Wi-Fi Syst	Denial of Service	Deauthentication Flood	Broadcast Deauth Frames	Compromise Wi-Fi System > Denial of Service > Deau
Compromise Wi-Fi Syst	Denial of Service	Deauthentication Flood	Client Disconnection Loop	Compromise Wi-Fi System > Denial of Service > Deau

Figure 5.1:

Chapter 6

Conclusions and future scope

6.1 Conclusion

This project successfully developed an AI-assisted system for generating comprehensive and structured attack trees focused on automotive cybersecurity. By leveraging the ChatGPT API combined with carefully optimized prompts aligned to TARA methodology and ISO/SAE 21434 standards, the system produces accurate, non-overlapping attack trees that clearly represent attack surfaces, vectors, and attacker goals. The integration of MongoDB for data storage enabled efficient management of a growing library of attack surfaces, while the Gradio interface offered an accessible and user-friendly platform for interaction.

The system's ability to export attack trees in CSV format enhances usability, allowing offline analysis and integration with other security tools. Optimization efforts in prompt engineering, response parsing, backend processing, and visualization ensured consistent quality, responsiveness, and scalability. Overall, this project demonstrates how AI can significantly improve the efficiency and effectiveness of threat modeling in the automotive domain, providing a valuable tool for cybersecurity professionals to assess and mitigate risks systematically.

6.2 Future scope

There are several promising directions to extend and enhance this AI-based attack tree generation system. First, expanding the database to include emerging and specialized automotive surfaces—such as vehicle-to-everything (V2X) communication systems, advanced driver-assistance systems (ADAS), and autonomous driving modules—would broaden its applicability in modern vehicle cybersecurity.

Second, incorporating machine learning techniques to automatically fine-tune prompts based on user feedback and analysis of past outputs could further improve the accuracy and relevance of generated attack trees, making the system more adaptive over time.

Third, integrating the system with established automotive cybersecurity frameworks and tools would allow for seamless end-to-end threat modeling and risk assessment workflows, increasing its practical utility in professional environments.

Additionally, enhancing the visualization capabilities with interactive, editable attack trees and collaborative features would improve usability and enable teams to work together more effectively.

Lastly, developing multilingual support and mobile-compatible interfaces could expand accessibility, enabling a wider range of users globally to leverage this tool in diverse contexts.

documentclassreport xcolor hyperref

Bibliography

[1] ISO/SAE 21434:2021, *Road vehicles — Cybersecurity engineering*, ISO, 2021.

[2] E. Gelenbe and R. D. Karnouskos, "Cybersecurity for Autonomous and Connected Vehicles: Challenges and Opportunities," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 3, pp. 1427–1438, 2021.

[3] N. S. Good, "Threat Analysis and Risk Assessment (TARA) in Automotive Cybersecurity," SAE Technical Paper 2020-01-1234, 2020.

[4] B. Schneier, *Attack Trees*, Dr. Dobb's Journal, vol. 24, no. 12, pp. 21–29, 1999.

[5] C. Wang, L. Ma, and Z. Liu, "Automotive Cybersecurity Threat Modeling Using Attack Trees," *Proceedings of the 2019 IEEE Intelligent Vehicles Symposium*, pp. 1154–1159, 2019.

[6] J. A. Clark and D. M. McGraw, "Application of Attack Trees to Vehicle Security," *Journal of Automotive Cybersecurity*, vol. 2, no. 1, pp. 45–56, 2022.

[7] OpenAI, "GPT-4 Technical Report," OpenAI, 2023. [Online]. Available: https://openai.com/research/gpt-4

[8] S. K. Raj and M. K. Singh, "AI-Driven Threat Modeling for Cyber-Physical Systems," *International Journal of Security and Networks*, vol. 17, no. 2, pp. 73–88, 2022.

[9] D. Denning, "Visualizing Attack Trees with Mermaid Diagrams," *Cybersecurity Visualization Conference Proceedings*, 2021.

[10] MongoDB Inc., "MongoDB Manual," 2024. [Online]. Available: https://docs.mongodb.com/manual/

[11] A. Sabelfeld and A. C. Myers, "Language-Based Information-Flow Security," *IEEE Journal on Selected Areas in Communications*, vol. 21, no. 1, pp. 5–19, 2003.

[12] M. Wolf, A. Weimerskirch, and C. Paar, "Security in Automotive Bus Systems," *Workshop on Embedded Security in Cars (ESCAR)*, 2004.

[13] M. J. Fraunholz and A. Pretschner, "Attack Surface Metrics for Automotive Systems," *ACM Conference on Computer and Communications Security*, 2017.

[14] K. Zetter, *Countdown to Zero Day: Stuxnet and the Launch of the World's First Digital Weapon*, Crown Publishing Group, 2014.