Identifying Cybersecurity Risks for a Drone Being Controlled Over a Webplatform for One or Several Domains

TONNY HUYNH AND ASHRAF ALZAIN
BACHELOR´S THESIS
DEPARTMENT OF ELECTRICAL AND INFORMATION TECHNOLOGY
FACULTY OF ENGINEERING | LTH | LUND UNIVERSITY

Bachelor’s Thesis

Identifying Cybersecurity Risks for a Drone Being Controlled Over a Webplatform for One or Several Domains

By

Tonny Huynh and Ashraf Alzain

Department of Electrical and Information Technology
Faculty of Engineering, LTH, Lund University
SE-221 00 Lund, Sweden

Abstract

Emerging technologies have increased the need for cybersecurity protection in web-based systems. This is especially true for drone systems, where any attack would threaten critical missions trying to save lives. In this study, the Remote Mission Control platform developed by Remote Aero is examined, the goal is identifying cybersecurity risks and proposing improvements to strengthen the system's overall security.

The methods used in this study included researching common cybersecurity threats, followed by an analysis of Remote Aero’s Remote Mission Control platform. This involved identifying both strengths and vulnerabilities in the code and system architecture through developer interviews and vulnerability assessments, compared against industry standards like the OWASP Top 10.

The analysis revealed some security weaknesses and strengths in the system. The strengths included the use of Role-Based Access Control and JWTtokens for session management. However, several security weaknesses were identified. Most importantly the system lacks Multi-Factor Authentication, which may make it vulnerable to brute-force and dictionary attacks. Additionally, there is no monitoring system implemented, which can leave malicious activities unnoticed like DDoS attacks. Also protection against CSRF and XSS attacks can be improved. Recommendations to improve security include implementing Multi-Factor Authentication, enhancing CSRF protections by implementing csrf tokens and integrating a real-time monitoring system to detect suspicious activity.

Future work should focus on securing the new SQL database, which was moved from Firebase to Supabase, creating potential risks for SQL injection attacks. To prevent this, it’s important to use parameterized queries and avoid raw SQL queries. Other suggestions include improving DoS attack protection with anomaly detection systems, dealing with emerging risks like GPS spoofing in drone operations, and adding Single Sign-On to make authentication easier and safer by reducing the need for multiple login credentials.

Acknowledgments

This Bachelor's thesis would not exist without the support and guidance of our thesis advisor, Dr. Sara Ramezanian, whose advice, feedback, patience, and expertise were crucial to this project. Without her help, this thesis would not have been possible. We also want to express our gratitude to our examiner Thomas Johansson, and Alexander Sandström at Remote Aero and his team for their incredible support and for providing the resources we needed to complete the thesis.

Tonny Huynh And Ashraf Alzain

Keywords: Drones, Remote Control, Websecurity, Cybersecurity, Webattacks

Contents

Preface

The division of labor in this thesis is detailed in Table 1, showcasing the responsibilities of each author who contributed to this thesis. The project began with both collaborators jointly discussing the direction of this study with Remote Aero's developer, such as its goals and the purpose and which department of cybersecurity that needs to be looked upon. This established the direction of the study and the authors chose the web domain of Remote Mission Control.

The responsibilities were defined by each collaborator's part in this thesis. Tonny analyzed the codebase in Chapter 4 and documented the results carefully which was then given to Ashraf in Chapter 5. The knowledge Ashraf possessed thanks to his expertise in the research field in the technical background helped form and establish the concluded analysis in Chapter 5. The remaining chapters' responsibilities were split depending on each authors time which effectively helped this thesis finish before the deadline.

Table 1: Division of labor in thesis

Section	**Responsibility**
Abstract	Ashraf
Preface & Ackknowledgements	Ashraf
Chapter 1 - Introduction & Background	Tonny
Chapter 2 - Technical Background	Ashraf
Chapter 3 - Methodology	Tonny
Chapter 4 - Results	Tonny
Chapter 5 - Analysis	Ashraf
Chapter 6 - Conclusion	Tonny

Chapter 1

Introduction

This chapter establishes an overview of the bachelor thesis, providing a background that introduces the objectives and research questions to the reader. Furthermore, this chapter also covers the limitation of this work and its purpose while also sharing the specific structure of the bachelor thesis.

1.1 Background

Several decades of research has been conducted on cybersecurity [1] with the focus on storing information securely. However, cybersecurity has been evolving to the point of battlegrounds no longer being limited to physical borders. With the expansion of the digital space, web applications are now more threatened than ever, affecting individuals, businesses and governments alike which introduces the need of websecurity. Other fields like healthcare, finance and education have also been deeply integrated into the digital cyberspace [2] , making cybersecurity on the digital infrastructure a central aspect of modern society.

Drone-technology is no exception as the introduction of Unmanned Aerial Vehicles (UAVs) [3] has demonstrated high potential, mostly in warfare where it has shown its most effectiveness, reducing the loss of countless human casualties. Beyond military operations, drone technology could be utilized in the health care sector, ranging from emergency operations such as delievering medicine for rapid transportation. All of this was made possible due to the emergence of recent technology such as artificial intelligence and the 5G network along with the need of reducing human involvement during the COVID-19 pandemic.

High-technology companies like *Remote Aero* are working to revolutionize this technology. However, with the expansion of technology comes the cost of adding extra security in order to mitigate the cyberthreats that threaten the system. In collaboration with Remote Aero, this bachelor thesis conducts a security analysis on the company's web platform that is called "Remote Mission Control". The purpose of this security analysis is to examine the company's existing system, with methods such as the identification of vulnerabilities and development of strategies to ensure drone operations remain protected against potential websecurity threats.

1.1.1 Remote Aero

Remote Aero is a remote Swedish company who specializes in developing a web platform for organisations such as sea rescue services, enabling them to operate drones safely and efficiently from a distance on a large scale. The platform "Remote Mission Control" was developed as part of a R&D-project as an experiment to establish a link between a webplatform and a drone in hopes to enable drone operations remotely in addition to sharing information with third parties. Currently, the platform does not save any data but works as a realtime-platform to operate drones. The platform hopes to simplify workflows, which enhances situational awareness for pilots who remotely fly drones. This feature helps them maintain high productivity while following regulations and safety protocols. Remote Mission Control is web-based and provides all sorts of access regardless of drone model, operating system, or hardware type. The program provides relevant information such as the drones position with its altitude, coordinates, and speed. Remote Mission Control also has the feature to simulate drones. In other words, the program operatates a simulated drone in a 3D representation of the real world, which from the user's perspective, behaves like a real drone but does not physically exist. This comes with its advantages which includes practical testing of drone operations without consuming valuable resources or dealing with real-world constraints such as battery life, weather conditions, or aviation regulation.

1.2 Purpose

The purpose of this thesis is to analyze the cybersecurity of Remote Aero's web domain when it comes to utilizing drones remotely in sea rescue operations. This remote process of controlling the drone through a webplatform represents a critical point of vulnerability, exposing the system to potential cyberattacks that compromises it from its intended purpose. These key components that could disrupt such functions include weak authentication mechanisms where the attacker gains control of the drone mid-flight. Additionally, threats such as Denial Of Service (DoS) attacks could also disrupt potential rescue operations, resulting in an unreliable system which could have severe consequences.

With this purpose in mind, this thesis aims to identify existing security protocols and evaluate whether these vulnerabilities exist, and provide recommendations to strengthen the system's protection against such potential cyber threats. Furthermore, the research will also explore Remote Aero's specific interests and requirements when it comes to emerging cybersecurity technologies, both to assess the technology applicability to their platform Remote Mission Control, and to find the tradeoffs between security and availability. This will ensure drone pilots that the security measures would not hinder their work in critical situations when it comes to overly complex authentication processes.

1.3 Goal formulation

The sea rescue service, who Remote Aero primarily collaborates with to develop the platform "Remote Mission Control", relies on critical applications of connected embedded systems such as IoT devices. These systems will be utilized by drones to locate and rescue people in distress when the product is ready. The first goal would be to enhance its security, an analysis of the system will be needed by evaluating existing security measures and provide improvement recommendations to the company.

The platform which is used to control drones and related systems is also used to transmit and collect real-time data through video and control commands, therefore it is particularly vulnerable to cybersecurity risks that can threaten its functionality and integrity. To enhance its security, a second goal would be an analysis of video streaming tools, exploring both WebRTC and SRTP, see Table 2: Glossary in Appendix A, Extended material.

These vulnerabilities could be exploited by for instance attackers that target weaknesses in encryption, authentication, and access control. Such attacks could result in data leaks or even manipulation of control commands, disrupting or hindering rescue operations. The third goal would be documenting and identifying such potential attacks.

Finally, the last goal would be researching the trade-offs between security and usability. This ensures that remote drone pilots can operate efficiently without obstacles caused by security measures.

1.4 Problem formulation

The thesis will investigate Remote Aero's current cybersecurity protocols, identifying both the strengths and vulnerabilities in the company's system. By researching their current system, it will be possible to explore protocols that could potentially improve their platform, leading to increased security of remote drone operations. These following research questions (RQ) will be answered throughout the report:

- RQ 1: What security measures exist in Remote Mission Control’s system and how effective are they against cyberthreats?
- RQ 2: What are the primary cyberthreats that target the web system and what vulnerabilities exist in Remote Mission Control’s system?
- RQ 3: What countermeasures can be implemented to mitigate these vulnerabilities that exist in this web-based drone platform? How can we enhance the system against the vulnerabilities?
- RQ 4: What are the security implications of implementing Single Sign-On (SSO) and how does it impact user experience?

1.5 Motivation for the thesis

Web security is a part of cyber security, a growing field that will increasingly play a key role in the expansion of the digital world [4]. As long as technology advances, so will the cyberattacks that threaten web platforms such as Remote Mission Control. The rising volume and cost of cyberattacks and data breaches [5] highlights the need for higher investment to mitigate web security risks and the early identification of vulnerabilities. Attackers are continiously attempting to develop new methods in hope to breach and disrupt systems like Remote Mission Control. Therefore, it is urgent to prioritize security, specifically web security in Remote Aero’s position due to the critical role that drones will play in the future. Both to prevent for instance, unauthorized access to the drones and command manipulation during flight operations.

This project and its efforts will provide competitive advantage if the authors would prefer staying in this cybersecurity matter in studies further or work. By conducting web security analysis on Remote Mission Control’s existing system, with methods such as identification of strengths and vulnerabilities of the system, this thesis will contribute to increased protection of drone operations against potential threats, thereby stengthening the security of the company’s platform. Additionally, it will also provide a valuable experience for the authors to challenge themselves, especially as this thesis requires practical and theoretical expertise in order to secure the web platform’s security. The customer’s confidence in continuing offering contracts and solutions for commercial and rescue-related purposes will also be maintained. This is given if the drones have the decisive functionality to locate and aid people in distress. This is done by identifying and possibly improving the platform that ensures that the drones and communication system are protected. This thesis will contribute once Remote Aero's research and development of the product is finished.

1.6 Limitations

To measure the security of a system, all departments and aspects related to the system must be considered. However, this thesis cannot identify all sorts of webattacks due to the limited time and resources of this bachelor thesis. For instance, an attacker that threatens the system only needs to exploit a single weak link to be able to compromise the entire system. The methods that strengthen the security measures against unwanted physical access to the drone are beyond the scope of this thesis. This includes lowering the risks for radio interference and jamming, leading to extra insurance of secure authentication to the backend systems. In other words, solely the cybersecurity of the web system will be taken into the analytics.

1.7 Disposition

The disposition of this report will be divided into six chapters that follow a systematic pattern. Each chapter is built upon the other and for the most readable experience, the authors recommend reading each chapter in order. Chapter 2 covers a technical background to the tools and concepts that will be mentioned in this thesis. The protocols and concepts that will be mentioned include common websecurity concepts such as the types of web-based attacks that can occur. Chapter 3 will clarify the methods that were conducted in this analysis, summarizing each phase that led to the results in Chapter 4. Chapter 5 will dig deeper into our thesis, reflecting on the results extracted from Chapter 4. Finally, Chapter 6 will close the conclusion of this project, summarizing the answers to the research questions, as well as highlighting opportunities for future work related to this research.

Chapter 2

Technical background

The chapter provides information on the technical standpoints for the reader, defining security protocols, languages, and computer science concepts that will be necessary to fully grasp the remaining chapters of this work. This includes exploring the concept of encryption, the definition of cybersecurity, and its aspects.

2.1 Web application

Web applications, such as Remote Mission Control, are just like any other ordinary application, except that they are web-based [4], meaning they utilize a web browser or a user agent such as Chrome or Microsoft Edge to access the web server. To run a web application, it can be implemented either on the server side using languages such as PHP or on the client side using JavaScript in the web browser.

2.2 Web security and web-based attacks

This subsection provides an overview of the web-based attacks which this thesis will be focusing on when it comes to the analysis of Remote Mission Control.

2.2.1 Sessions

Sessions refer to the period during which a user interacts with a website [6]. They maintain state information related to the user's interaction, such as preferences and authentication status. The server initiates a session when the user visits the website, and it remains active until the user logs out or the session expires.

A unique session ID is generated to identify and track the session. This session ID is typically stored in cookies or as a parameter in URLs, allowing the server to associate user requests with their respective sessions. This ensures a personalized experience throughout the user's visit.

2.2.2 Cookies

Cookies are small text files that track a user's session and identify users on a website [6]. When a user visits a website, the server sends a cookie to the web browser containing a unique identifier (like a session ID). The cookie stores information such as session details, login status, and user preferences like language settings. Cookies can for example be then stored locally in the user's web browser. The next time the user visits the site, the browser sends the cookie back to the server, allowing the website to remember the user’s session and settings.

2.2.3 Cross-Site Scripting (XSS)

Cross-Site Scripting is a web security vulnerability that occurs when an attacker injects a script into a user's website, typically written in JavaScript [7]. The script executes on the victim’s website without the user’s knowledge, often with the goal of stealing cookies, sensitive login information, or redirecting the user to another website.

XSS attacks can be categorized into two main types: persistent and non-persistent. A persistent XSS attack occurs when an attacker injects malicious code into a website, where it is stored and executed every time a user visits the page. The attack can be used to redirect them to a fraudulent website designed to steal session cookies or other sensitive data.

In contrast, non-persistent XSS attacks do not involve storing the injected code permanently. Instead, the attacker sends the victim a link containing malicious JavaScript code. When the victim clicks on the link, the script executes which can potentially lead to stealing cookies or hijacking the session.

To prevent XSS attacks, it is crucial to validate user input in web applications to ensure that malicious scripts are detected and blocked from execution [7]. A PHP function, htmlspecialchars(), can be used, for example, to translate characters to their respective entities to prevent unwanted code execution. Similarly, the function strip_tags() can be used to strip all HTML tags from a string, reducing the risk of script injection. In addition, to minimize the risk of cookies being stolen by JavaScript, websites can restrict access to cookies by enabling the HttpOnly flag in session management.

While validating user input is an important step in preventing XSS attacks, it is not enough on its own. Controlling which sources are allowed to execute code is equally critical, as many XSS attacks rely on running scripts from external sources rather than the legitimate website. One effective way to address this is through a Content Security Policy (CSP), which restricts the execution of code from untrusted domains. CSP is implemented via an HTTP header that specifies which domains are permitted to run scripts.

2.2.4 Cross-Site Request Forgery (CSRF)

A CSRF (Cross-Site Request Forgery) attack occurs when a hacker tricks a website into believing that a request made by the hacker was actually made by the legitimate user [7]. This happens because websites often use cookies for authentication, which are automatically included in requests sent from the user's browser.

For example, if a user is logged into their bank account, a hacker might trick them into visiting a malicious website they created. Without the user realizing it, this website can secretly send a request to the bank, such as transferring money to the hacker's account. Since the user is already logged in, their session cookie is used in the request, making it appear as if they approved the transaction, even though they had no involvement in it.

To protect against CSRF attacks, using secret validation tokens is essential [7]. These tokens require each request to include a unique value that cannot be accessed by an attacker, such as a session ID or a randomly generated token. This ensures that only legitimate requests from the user are executed.

Another effective defense mechanism is checking the Referer header. When a browser sends a request, it often includes the URL of the page where the request originated. The server can then verify if the request comes from the same website before executing it.

There are two ways to implement Referer validation:

- Lenient Referer validation – The server accepts requests even if the Referer header is missing.
- Strict Referer validation – Rejects any request that does not include a Referer header.

2.2.5 Session hijacking

During a session hijacking attack, a hacker gains access to the session ID of the victim, allowing them to take control of the session [6]. The attack can be carried out in different ways:

- Session prediction – The adversary guesses the session ID, which is possible if the ID is not sufficiently randomly generated and instead generated by incrementing a value, for example.

- Session sniffing – This occurs when an adversary eavesdrops on the network communication to obtain the session ID. HyperText Transfer Protocol (HTTP) requests that are not encrypted send cookies in an unencrypted form. If an attacker eavesdrops on the traffic, they can steal the cookies and hijack the session.

The best prevention for this attack is encrypting all network communications by using SSL/TLS (Secure Sockets Layer/Transport Layer Security) throughout the entire session.

2.2.6 Denial-of-service (DoS) and Distributed-Denial-of-Service (DDoS)

Denial-of-Service (DoS) [8], is a type of attack that floods system resources with traffic, resulting in an unresponsive system that cannot respond to the requests sent by the user. DDoS Distributed-Denial-of-Service (DDoS) is a variant of the DoS attack on a larger scale, where the attacker shuts down the entire system for the user by overwhelming it with a massive number of requests.

To ensure protection against DoS attacks, systems need to incorporate the Intrusion Detection System (IDS) protocol or a DDoS protection product due to the difficulty of differentiating between fake traffic and real traffic [9]. Another method is increasing internet bandwidth to lower the likelihood of DoS attacks overwhelming the system.

2.2.7 Man-in-the-Middle attack (MitM)

A Man-in-the-Middle attack (Mitm) [8], is a type of threat that occurs when an unauthorized third party intercepts the communication between the server and the client. The attacker mimics both the server and the client, making them think they are communicating with each other while enabling access to steal, eavesdrop, or modify the transmitted data.

The most common MitM attacks are usually directed at real-time communication, transactions, or data exchanges, where MitM types typically include some form of hijacking or IP spoofing. However, protection can be provided by installing a detection system that alerts if the network flow is disturbed. Additionally, adding further security layers to Wi-Fi with the help of a virtual private network will also provide extra protection against MitM attacks.

2.2.8 Time-Memory-Trade-Off (TMTO) attack

A Time-Memory Trade-Off (TMTO) attack is a technique where an attacker speeds up the process of an attack by performing a trade off between time and memory [10]. The goal of this type of attack is to break the one-wayness of the hash functions and recover assets such as passwords from their hash values. A common TMTO attack utilizes rainbow tables, which involves precomputing several tables of compressed hash values and their corresponding inputs using several reduction functions. This reduces memory usage significantly compared to storing every possible hash input-output pair. By the end of the attack, the attacker can quickly search up the target hash in the table and recover the original asset, eliminating the need for expensive brute-force attacks. This reduces time usage significantly.

2.2.9 SQL injection attack

To store, retrieve, and write to a database, a computer language like Structured Query Language (SQL) is needed [6]. SQL injection works as a manipulation of the database at the backend by inserting malicious commands that may result in the deletion of or unauthorized access to information in the database.

An SQL injection attack normally exploits vulnerabilities by taking advantage of the lack of proper validation in the database [6] . One method to prevent this is to implement input validation that checks for malicious user inputs. However, this approach is quite challenging due to the wide range of possible valid and invalid inputs.

The best method is the implementation of a Web Application Firewall (WAF), which utilizes security measures such as signature identification and IP reputation analysis—a method used to determine the trustworthiness of a user by analyzing past behavior and activity on the internet—to prevent such attacks.

2.3 Application programming interface

An Application Programming Interface (API) is a set of rules that allows different software components to communicate with each other. It acts as a bridge between applications, enabling them to send and receive data in a structured way [11].

In a web application, an API connects the frontend (user interface) with the backend (server and database). For example, when a user submits a login form, the frontend sends a request to the backend through an API. The backend then verifies the credentials and responds with either access approval or an error message.

2.4 OWASP

Open Worldwide Application Security Project, shortened to OWASP [11] is an open source project built to guide organizations to achieve security across its platforms. OWASP ranks these 10 security flaws based on how critical they are to web application security. Here are the 10 security flaws listed by OWASP:

1. Broken Access Control
    - a. Flaws in the permission policy when it comes to handling which user has access to what. For instance, a default user should not have the same capabilities as the administrator.
2. Cryptographic Failures
    - a. Flaws in the encryption, leading to worse protection for assets such as passwords, personal information, business secrets and health records.
3. Injection
    - Security flaws that allow attackers to inject code which is malicious and could lead to manipulation of databases or commands not executing as intended.
4. Insecure Design
    - Vulnerabilities which are caused by design and architecture, rather than implementation issues.
5. Security Misconfiguration
    - Not proper security settings, such as missing configurations related to cloud permissions.
6. Vulnerable and Outdated Components
    - Untracked components of a system can lead to a vulnerable security system due to components being outdated or not having the right compatibility when issues arise.
7. Identification and Authentication Failure
    - Flawed authentication systems that could be exploited by malicious actors, such as weak password policies.
8. Software and Data Integrity Failures
    - Protection flaws against unauthorized modification due to the relation between the application and the untrusted plugin.
9. Security Logging and Monitoring Failures
    - Inproper system for detection and monitoring which allows security breaches to slip through the defenses of a system, resulting in the difficulty of a response for an attack.
10. Server side Request Forgery (SSRF)
    - Fetching external resources without a validation towards the provided URL, enabling attacks towards internal systems.

2.5 Authentication

An authentication system is a security mechanism which protects assets from non authorized-users who hope to gain access to specific resources such as applications or data [10]. In a websecurity context, it is designed to verify certain individuals through a verifying system which consists of a user utilizing credentials such as a username and a password. If the credentials are confirmed to match with the database, the authentication is successful and the user gains access to the resources which the authentication system is protecting. However, to ensure non-unauthorized access, various strong authentication methods and practices have been developed to combat the malicious actors from infiltrating the systems. These policies include strong passwords policies and methods such as MFA.

2.5.1 Strong password policies

A strong password policy is the most common practice to prevent unauthorized access. By requiring passwords which are unique, complex, and long, attackers will have further difficulties to infiltrate a web system which has an authentication system. Some policies are as listed below:

- Minimum length (e.g., at least 12 characters).
- Rotate password regularly (e.g., at least once a year).
- A mix of uppercase and lowercase letters, numbers, and special characters.

Other than this, most authentication systems do not store the password in plain text in the database, instead they are hashed using strong cryptographic algorithms which protects passwords even if a database is leaked or compromised.

2.5.2 Multi-Factor Authentication (MFA)

Multi-Factor Authentication (MFA) has become one of the most effective methods to combat TMTO attacks. This practice involves enforcing a system to provide further forms of verification other than the regular credentials login system [10]. Some factors include:

- Credentials such as password or PIN.
- Time-based one-time passcode which expires shortly after being deployed.
- Biometric data such as facial, Iris and finger recognition

Some websystems combine all three of these factors in order to reduce the likelihood of unauthorized access.

2.5.3 Salt

Salt is a unique random value added to each password before it is hashed and can be stored with the hash value. This ensures that even if two users have the same password, their stored hashes will be different. Salt makes it harder for attackers to use rainbow tables, pre-generated lists of common passwords and their hashes, to crack passwords [9]. It also ensures that identical passwords produce different hashes, making it harder for attackers to spot users with the same password.

2.5.4 Pepper

Pepper is a secret value added to passwords before hashing, but unlike salt, it is not stored in the database. Instead, it is stored in a separate secure location or not stored at all [9]. This extra layer makes it more difficult for attackers to reverse hashes, even if they gain access to the database.

2.6 Single Sign-On (SSO)

Single Sign-On (SSO) is a powerful authentication technology which makes it possible for users to log in once and access multiple systems or applications without the need to authenticate separately for each service [12]. SSO utilizes a central authentication server [13] which issues tokens or credentials after the initial login. This allows the user to access all connected applications without authenticating separately again. Additionally, due to this feature, it is a solution which enhances UX and improves the security by reducing the amount of passwords needed and the frequency of authenticating.

Chapter 3

Method

This chaper provides the insights to the various phases of the thesis, including the strategies employed to successfully complete the task given by Remote Aero. Furthermore, to approach this assignment effectively, it is essential to have a solid understanding of websecuriy to be equipped to fully analyze Remote Mission Control's system. Therefore, the methodologies adopted by the authors consist of researching and collecting data from quality and trustworthy sources and materials such as peer-review academic journals and books. A thorough analysis of the code of Remote Aero will be conducted to identify potential strengths, weaknesses and vulnerabilities in terms of security of the system.

3.1 Phases

The first phase initiated with researching the topic of *cybersecurity* and its subcomponent *websecurity*, followed by reviewing the security structure for Remote Mission Control by analyzing the source code. The authors of this study also had a parallel course, Computer Security which provides guidance to the thesis in topics such as authentication and attacks that are applicable to the web. The next method involves understanding the whole architecture followed by doing a vulnerability scan to identify potential gaps in Remote Aero's system that the threat actors could exploit. This includes evaluating the system from potential individual attacks. Lastly, providing recommendations and studying emerging technologies that Remote Aero could implement which might strengthen the system. Regular meetings with the company's web developers and interviews were also conducted, ensuring good verbal and written communication throughout the work of the thesis, which serves as a good method for moving the project forward.

Here are the list of different phases of the project which the authors went through:

- First Phase
  - Research on the topic of websecurity
- Second phase
  - Reviewing security structure of Remote Mission Control
- Third phase
  - Vulnerability identification

- Fourth phase
    - Recommendations

3.2 First phase: Research and data collection

To successfully complete the first phase of this thesis, a collection of data regarding the latest websecurity practices, protocols and technologies has to be applied. This data is crucial as it requires good knowledge for understanding the potential vulnerabilities of a system and assessing the potential threats it may face. This involves conducting studies on relevant material through high-quality sources such as Google Scholar and the library of Lund University. Fundamentals which provides minimum requirements that need to be understood in order to fully complete the cybersecurity analysis of the web application. The data and research which the the authors collects includes mostly various types of web-based attacks, including:

- Cross-Site Scripting (XSS)
- Cross-Site Request Forgery (CSRF)
- Session Hijacking
- Denial-of-Service (DoS) and Distributed Denial-of-Service (DDoS) Attacks
- Man-in-the-Middle (MitM) Attacks
- Time-Memory Trade-Off (TMTO) Attacks
- SQL Injection

Furthermore, the parallel course *computer security* also helps enrich the studies by introducing basic security concepts and relevant protocols.

3.3 Second phase: Code and architecture analysis

The second phase consists of performing an architecture and security analysis of Remote Mission Control. The objective is to assess the current security protocols, codebase, and architecture of the system, detailing what security measures it provides for the platform. This is done by conducting an initial interview with the developers of the company, gaining insights into the company's current security measures. This serves as the first step of gathering information which helps underlying potential vulnerable parts of the system. Furthermore, in order to gain access to the codebase, signing a non-disclosure agreement (NDA) is required.

Engaging in discussions with the company's lead developer further strengthens this thesis analysis, ensuring that the thesis is keeping up with the company's demands and wishes for the security requirements and expectations. This is done by regularly updating the status of the thesis, as well as validating security concepts and the company's implementation in the codebase. Ensuring that the authors have clear communication with the developers and preventing misunderstandings such as difficulties in analysis of the code which might impact the accuracy of security analysis.

The process of the architecture assessment begins with analyzing the Backend as it contains the system's most critical components, extracting data such as the web application's current security measures and how the communication is being handled between each class. This is done systematically because focusing on how different components interact and how the security measures are deployed, helps the analysis pinpointing which areas that the architecture assessment should focus on. For instance, by determining the communication pattern, Remote Mission Control's data flows, and its access controls, the assessment can identify potential security risks, such as unauthorized access to the drones due to weak authentication mechanisms.

3.4 Third phase: Vulnerability identification

Once the architecture assessment is finished, the next phase involves analyzing the collected data. This collected data contains key details about the web application's infrastructure, such as its components, its potential vulnerabilities, and its security mechanisms. Details which are then examined carefully to identify the strengths and the weaknesses of the websystem.

Additionally, the extracted data is then assessed and compared against the common web-based attacks which this thesis covers, picked from OWASP Top 10. The reason is to determine whether the system has any known vulnerabilities which these attacks vectors could exploit. In other words, by examining the system towards these attacks, the authors can determine how well the system is protected against them.

3.5 Fourth phase: Recommendations

Finally, once a complete analysis is conducted, a set of recommendations has to be provided to Remote Aero, which is displayed in this thesis. However, for research ethics purposes, Remote Aero received this thesis earlier before the official publication for the company to fix and implement the security measures. These recommendations are based on the vulnerabilities that the authors have found within the scope of the thesis, which includes best practices and security enhancements aligning with industry-standard frameworks provided by for instance OWASP.

3.5.1 Emerging technologies

Once finishing the recommendations for implementing security measures, the project is aiming to explore emerging technologies that the company, Remote Aero, requires for future projects in order to expand Remote Mission Control. These emerging technologies include:

- Single Sign-On (SSO) Evaluation:
    - Assessing the implementation of Single Sign-On (SSO) as a potential solution for secure and seamless user authentication.
- Trade-Off Analysis between Security and Usability
    - Assesing the right balance between securing the platform and maintaining a smooth UX for remote drone pilots.

3.6 Documentation and reporting

To document findings for this report along with the analyses, and recommendations we log and collect all data in parallel with the project's ongoing activities. For instance, by analyzing the code along with searching for web vulnerabilities, the writing became effective as this approach allows for immediate recording of detail and accurate information.

The process of documentation starts with the gathering of information regarding the methods and subject of this thesis provided by Remote Aero. This includes background information, goal- and problem formulation as well as motivation for this thesis. Additionally, the limitations are also explored in the early process of the documentation.

The second part of the documentation is focused on finding Results and the code analysis. Technical background is also provided while focusing on the Results. However, this proved to be ineffective as the result part is changing dynamically after each supervision. This led to the technical background being handled after nearly completing the result part.

The last part of the report is focusing on summarizing the findings, analyzing it and discussing the conclusion. This last part also includes future work while also reflecting the ethical part of this report.

3.7 Source criticism

The sources provided in this thesis mostly originated from LU discovery, where most of the sources are well established as LU discovery is part of Lund University's library, a university standing in the top 100 of the international ladder. Google Scholar is also used as a source to find reliable sources to support the authors theories and analysis. This search engine is recommended by Lund University and from the authors research, Google Scholar categorizes the journals, articles or studies based on where the study or research was published, who it was written by, and how recently it has been cited in literature related to scholars. Trusted and well-known academic books, authored by highly esteemed professors and researchers are also used as sources in this thesis.

3.8 Web application tests

Prior to this thesis, the authors collaborated with the company on an earlier project whose aim was to develop a 3D simulation of this drone application. For further details, see Appendix A, Extended material, Figure 1, which displays a drone in Remote Mission Control's simulation mode. This development provided valuable insights that inform the current research.

Chapter 4

Result

This section provides the outcomes of the thesis work, showing results of the analysis of the codebase and identification of existing security measures. Additionally, an identification of vulnerabilities will be looked upon. Lastly, studies of Remote Aero's suggested emerging technologies will be conducted.

4.1 Architecture

Remote Aero has structured its web application into two main components, the Backend and the Frontend. All core functionality lies in the backend, this includes security features such as the user authentication and the communication between the web application and the drone.

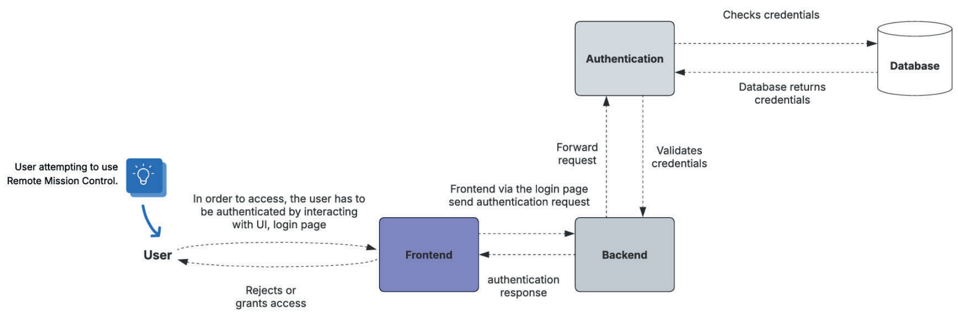

Figure 1: Authentication in login page, returns an access token

The frontend acts as a gateway allowing users to interact with the system through the user interface (UI). A key component of the frontend is the login page, which authenticates users by communicating with the backend database to verify their credentials. Figure 1 illustrates the data flow involved when a pilot authenticates their identity. Upon successful authentication, the user is assigned a predefined role and an access token. The user's role determines the functionalities available to the user within the web application. For instance, a *pilot* user will be able to perform drone operations while a *spectator* user will simply observe the operation without interference. To launch the drone, invoking relevant API endpoints is important, such as those that updates the drone's telemetry which includes its status such as its flight path on the map. Additionally, these APIs transmit other data from the Backend, allowing Frontend to access crucial data via requests such as wind speed, battery status, flight speed, traffic data via ADS and AIS, and the drone's gimbal angle.

The backend of the web application communicates with the drone using a Raspberry Pi that is built into the drone. It sends and receives data during flight with the help of external services, such as the 4G network, as shown in Figure 2, which illustrates the communication between the system and the drone.

The drone maintains a constant connection using WebSocket requests and transmits live video to the pilot through WebRTC. The backend receives this data and forwards it to the frontend, specifically to a component called scrubbing. This component allows the pilot to view the drone's surroundings in real time, using video captured by an onboard camera.

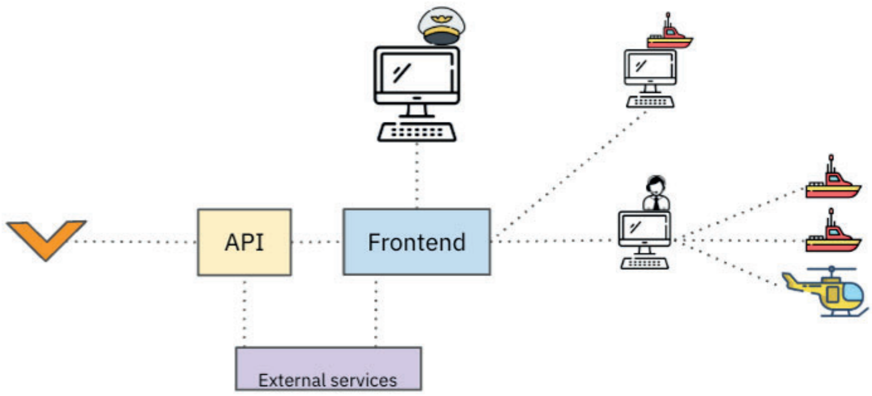

Figure 2: Communication between the system and the drone, utilizing APIs and external services (Provided by Remote Aero)

To summarize the architecture, each key component relies on the security of the authentication system. Before any component is authorized to communicate with the Backend or access any assets, the user must go through the static authentication and gain an access token. This ensures that only authorized users can interact with the web application.

4.1.1 Risk assessment

After examining the general architecture of Remote Mission Control, this thesis finds it necessary to assess the damages that cyber threats may cause based on the potential attack surface, the impact they would cause and the most probable attack scenarios. The damage assessments are measured in levels as follows:

1. High - severe consequences if an attack is successful
2. Medium - causes moderate consequences if an attack is successful
3. Low - causes minimal consequences if an attack is successful

Table 2: Damage assessment

Attack surface on the Web application based on findings	**Damage Assessment**	**Attack vectors (examples, could or could not happen)**
Authentication & authorization	High	Brute-force attacks, dictionary attacks, session hijacking
Backend & API-security	High	Injections, insecure endpoints
Network & communication	Medium to High	MitM attacks
User Interface (UI)	Medium to High	Cross-site scripting (XSS), CSRF
Logging & Monitoring	Medium	Log manipulation

The damage assesment in Table 2 is based on the assumption of where the attacker will gain the most control or cause the most damage if an attack is successful. For instance, by attacking the authentication mechanisms, the attackers will gain full access to the web application, allowing the attackers to bypass the authorization and compromise the system however they like. This includes the risk of sabotage or manipulation of flight operations, which can lead to serious consequences. Similarly, threats that target the Backend and the APIs would enable the attackers to control core functionalities, marking another high-risk scenario.

Protecting the communication between the web application and the drone is essential and should be carefully reviewed in case of a cyberattack. For instance, strengthening the network configurations could protect the system from MitM attacks which allows attackers to intercept or manipulate the data in transit. This data includes drone commands or data for live streaming, assets which enables the user to view the drone's surroundings. Therefore, this thesis has given this classification as medium to high as this risk is not quite on the same levels as breaching the authentication and the Backend system.

The Frontend is where the UI is constructed and could potentially be exposed to XSS attacks, which allow attackers to steal credentials or execute unauthorized actions. Such attacks do not immediately grant access to backend systems. However, it can be utilized to gain further control of the system, making the risk medium to high depending on the attacker's motives.

Logging and monitoring mechanisms are essential for early detection of ongoing attacks. For instance, cyberthreats such as log manipulation can hide malicious activity, causing difficulties for the system to detect the attack. Furthermore, these kinds of attacks do not directly target the system. However, failing to detect security breaches impacts the other parts of the system, making this medium risk.

By dividing the general system into distinct categories and prioritizing them based on the potential damage an attacker can cause, this thesis adopts a more structured approach. This ensures that the most high-risk areas receive the highest priority in efforts regarding websecurity.

4.1.2 Attack scenarios

This subsection will give insights into various types of attack scenarios which an attacker might utilize based on the risk assessment in Section 4.1.1. However, not all attacks will be considered as they are beyond the scope of this thesis.

4.1.2.1 Authentication and Authorization

1. Attacks on the authentication system
    a. Scenario: Remote Mission Control is operating a critical drone mission. However, a threat actor identifies this system as a target and attempts to disrupt the operation by seeking unauthorized access. The attacker navigates to the web browser and enters the domain of Remote Mission Control, noticing an authentication system which is blocking

the attacker from gaining access to the operating drones. However, given the value which the attacker gets of compromising the drone mission. The malicious actor decides to attempt bypassing the security system by employing various types of cyber attacks:

i. Brute force attack:
    1. This type of attack is complex and often involves trying a large number of credential combinations before the correct password is found. For instance, the attacker can approach this method by automating the attacks by utilizing bots.

ii. Dictionary attack
    1. The attacker utilizes a list of predetermined passwords which the malicious actor possesses from identifying the most common passwords in databases or previous database leaks. By systematically attempting these passwords, the attacker can increase the likelihood of succeeding and gaining access to an authenticated account.

iii. TMTO attack
    1. After obtaining hashes from the database of Remote Mission Control, the attacker decides to utilize a rainbow table to crack the password hashes. This is done by iteratively reducing the hash values to potential plaintext passwords until a match is found of the obtained hash values.

2. Broken access control
    a. Scenario: A low privileged user unintentionally alters the course of the drone operation during flight due to Role-Based Access Control (RBAC) being improperly configured, allowing the lower privileged users access controls beyond their permissions. An attacker could seize this opportunity by targeting the lower privileged users, having a wider pool of targets to select in order to successfully sabotage drone operations.

4.1.2.2 Backend & API security

- Injection attacks
    - Scenario 1: Malicious actors attempt to exploit input vulnerabilities, starting by injecting malicious commands into the web application's login system through NoSQL injection attacks, see Algorithm 1, where an attacker attempts an injection attack. The purpose of this is to fetch the username and password without providing legitimate credentials. This is possible by sending query payloads through the authentication system. This happens

because databases often execute dynamic queries based on user input in order to fetch data.

NoSQL query:	db.users.findOne({ "username": InputUsername, "password": InputPassword });

Algorithm 1: NoSQL injection

- Insecure Endpoints:
    - Scenario: The malicious actor analyzes the architecture of the web application, finding and targeting an unprotected API endpoint or a misconfigured authentication mechanism within the system. This enables the attacker to fully exploit that weakness, bypassing the authentication and gaining critical data. Typical attacks on such endpoints include:
        - Attack 1: Data tampering
            - Attackers manipulating API-requests to tamper with data, such as injection attacks or unauthorized modification.
        - Attack 2: DoS
            - The attacker overloads the unprotected API with a high volume of requests.
        - Attack 3: Unauthorized access
            - The attacker exploits the absence of authentication on the API.

4.1.2.4 User Interface

- Scenario 1 : The attacker decides to find vulnerabilities of the system regarding whether or not the system lacks input validation.
    - XSS
        - Persistent XSS
            - The attacker injects malicious scripts into the web application's database, permanently storing them dormantly until a user triggers this script. For instance, after a pilot finishes its flight operation, the pilot might leave a status report which will be read by another pilot in the future. By opening

this report, the attacker can steal the session cookie from the authenticated pilot who aims to read that specific report without notice.

- Reflected XSS
    - Similarly to the previous XSS attack type, the attacker targets an authenticated pilot by sending an email with a link, containing malicious scripts. By opening this link, the attacker can execute the script, stealing authentication tokens, session ID or any other valuable assets which can cause damage to the web system.

- Scenario 2:
    - CSRF
        - The attacker tricks the authenticated user into unknowingly executing unintended drone commands through the user's active session, bypassing the authentication system.
            - This occurs through a third-party website or email which the attacker embedded malicious requests. If the user interacts with this source through links or other forms of request, the attacker could execute commands using the user's active session. This occurs due to the system seeing this request as legitimate, as the command originates from the user's browser.

4.1.3 Existing security measures

The authentication class is the primary security measure of the platform and is designed to ensure access control throughout the web application. The purpose of the authentication class is to ensure that there is no unauthorized entry to the program. It also contains inner nested classes, where each subcomponent is used for different authentication tasks, such as defining user roles, as shown in Algorithm 2. Furthermore, there is no registration functionality, meaning only selected individuals have access to the system, lowering the risk of insider attacks.

The next security measure Remote Aero has enforced is the introduction of user roles. The purpose of this is to determine which action a user can perform within the system depending on the user's role. For instance, a user who wants to view a flight operation should not be able to disrupt the pilot user, who is performing critical tasks such as rescue operations, which mitigates insider attacks.

```
RBAC Code:

Class Role(Enum):
Pilot = a
Spectator = b
Drone = c
Launcher d
```

Algorithm 2: RBAC Code

The third security measure applied in the web application is the implementation of a backoff-algorithm as displayed in Algorithm 3. This results in a progressive time delay increase for each failed login attempt on the authentication system. The maximum attempts a user can try to login is five, increasing the difficulty for the maliscious actor to perform password attacks such as TMTO, dictionary, or brute force attacks.

```
FUNCTION handle_backoff(username):
    IF username exists in login_attempts:
        IF login_attempts[username].count >= MAX_ATTEMPTS:
            time_since_last_attempt = CURRENT_TIME - login_attempts[username].last_attempt

            IF time_since_last_attempt < BACKOFF_TIME:
                RAISE "Too many login attempts. Please wait" (HTTP 429)

        INCREMENT login_attempts[username].count
        UPDATE login_attempts[username].last_attempt to CURRENT_TIME
    ELSE:
        CREATE new LoginAttempt for username with count = 1 and last_attempt = CURRENT_TIME
```

Algorithm 3: Backoff-handler pseudo code

After the user is successfully authenticated, the application manages the session using JWT (JSON Web Tokens). As it is shown in Algorithm 4, these tokens are signed with the HS256 algorithm, which is a type of HMAC (Hash-based Message Authentication Code). HS256 uses SHA-256, a modern and widely trusted hashing method approved by NIST. This standard is commonly used in today's security systems. These tokens are time-limited which means that the user must re-authenticate between short time-intervals to ensure the authentication of the user is correct. Furthermore, the authorization uses the framework OAuth2.0 for ensuring access control securely, implemented with FastAPI and websocket is used for real time communication.

```
FUNCTION create_access_token(user):
    GENERATE JWT payload with:
        - token_type = "access"
        - username
        - roles
        - expiration timestamp
    RETURN JWT signed with HS256 and JWT_SECRET
```

Algorithm 4: Pseudocode of generating an access token

4.1.3.1 Existing countermeasures

Table 3 displays the attack vectors listed in subsection 4.1.2 and the current existing countermeasures outlined in subsection 4.1.3.

Table 3: Attack scenarios in Subsection 4.1.2 and how Remote Aero protect against them

Attack scenarios	**Existing countermeasures**
Attacks on the authentication	Backoffhandler, hash functions using SHA-256,
Exploits of broken access controls	RBAC is carefully integrated. For instance, drones can only be accessed by pilots.
Injection attacks	The database, firebase does inherently protect against injections

	targeting databases which rely on user inputs as queries.
Insecure endpoints	All APIs require authentication before use/communication
XSS, CSRF	See Section 4.5

4.1.4 Database

The company uses Firebase, a web-based NoSQL database that doesn't rely on traditional SQL queries, reducing the risk of SQL injection attacks. Firebase database offers advantages as they require strict rules for individuals and businesses using its services. These rules are highlighted in Firebase's setups guide which ensures secured data access control for authorized users who attempt to read or modify data.

Furthermore, these rules can be seen as built-in regulatory security measures, meaning that organizations such as Remote Aero that uses Firebase's services simply need to complement with additional security measures rather than building the security from scratch. As for Remote Mission Control, the company has implemented that only users with the correct roles can write and read to the database as displayed in Algorithm 5, the source code implemented by the developers. Additionally, firebase has built-in basic monitoring and logging features, making firebase an excellent choice for organisations to track and respond to threats in their early development.

```
rules_version = '2';
service cloud.firestore {
  match /databases/{database}/documents {
    match /{document=**} {
      allow read: if request.auth != null;
    }
  }
}
```

Algorithm 5: Source code for Firebase rules

4.2 Interview

The authors of this study have conducted an interview with the company and the corresponding answers (A) to the following questions (Q) are listed below:

- Q: What are the most effective security measures implemented in the web application?
    - A: The most effective security measure implemented in web applications is the authentication system. The security is currently custom built. However, The company is migrating to Supabase, implementing email-based authentication on the backend, and handling credential resets as well as verifying passwords, access tokens, and refresh tokens. Recently, X509-certificates have been implemented to strengthen the websecurity and there are plans for the SSO protocol to be integrated to authenticate organisations.
- Q: What potential vulnerabilities exist within the system, and how are they mitigated?
    - A: There are not any known vulnerabilities other than that the security system is custom built, no known security standards were followed during the design of the system. Furthermore, the cloud operator used in the external services for the web application lack DoS-protections.
- Q: How does Remote Mission Control automatically detect and respond to cyber threats?
    - A: The system lacks automatic detection of any cyberthreat. However, there are log activities through the backend which regularly checks all access and attempts to the APIs of the web system.
- Q: What is the company's incident response plan in the event of a cyberattack?
    - A: The drones in operation have redundant connections over VPN which does not go through the cloud. In the event of a cyber attack whenever the attacker accesses a user account during a flight and Remote Aero notices it, the company can then disable the cloud access, and switch entirely to the redundant connection.
- Q: What's the plan for the SSO functionality?
    - A: Enhancing user experience by for instance, adding a feature where logging into the web application using google authentication is possible.
- Q: Is weblisting disabled?
    - Environment variables are included in the JavaScript code that runs, allowing API-keys, etc., to be retrieved from there. However, the variables are restricted as they can only be retrieved and used from specific IP-addresses, such as rmc.remote.aero. Also, while

    Remote Aero has environment files, the variables cannot be accessed from any URL in the backend.

- Q: Does Remote Mission Control have any protection against XSS?
    - I don't think that we have any explicit protection against XSS attacks.

Key points from the interview is that the current websecurity system is custom built and does not follow any standards recommended by organisations such as the NIST or OWASP. Remote Mission Control is relying on external services and the collaborated cloud operator does not provide any DoS-protections. There is no automatic detection in the event of a security breach, requiring an implementation of an anomaly detection in the near future. These factors alone pinpoint the need of getting further resources to implement these necessary features for web security.

4.3 Vulnerability identification

For vulnerability identification, the thesis will utilize OWASP top 10 security flaws. Furthermore, the authors have decided to pick three of those flaws for vulnerability identification on Remote Mission Control as for the rest are beyond the scope of this Bachelor thesis.

4.3.1 Identification and authentication failure

According to OWASP, the requirements placed in Table 4 represent minimum security measures to prevent attacks related to authentication. The drones operating remotely need secure authentication to be able to withstand attackers from taking control of the drone. If these criterias are not met, the system might suffer from weak links which the attacker could exploit. Table 4 represents the evaluation whether Remote Aero meets these criterias [11].

Table 4: Checklist for Authentication security

Security Requirements	**Check? (Y/N)**
Not allowing automated brute-force attacks	✅ Y, backoff-handler
Not allowing weak passwords such as '1234'	❌N

Hashing passwords	✅ Y
Providing Multi-Factor Authentication (MFA)	❌N
Not allowing ID to be exposing in URL	✅Y
Not reusing old session IDs	✅Y
Ending sessions properly	✅Y

From the findings in Section 4.1.2, the analysis indicates that Remote Mission Control is missing two critical areas. Multi-Factor Authentication and allowing weak passwords. These failed areas such as weak passwords allow threat actors to perform brute-force attacks, a technique which utilizes for instance systematic attacks to find the password combinations needed to access the system. However, as said before, Remote Mission Control has a backoff-handling system which progressively delays the time to attempt the login if the authentication fails.

Furthermore, only maximum 5 login attempts are allowed, resulting in brute-force attacks being incredibly difficult to perform. But, this also leads to a potential scenario where a DoS attack is possible, especially when the user attempts the authentication and the attacker could lock them out [11]. For further protection against brute force attacks, adding a CAPTCHA for the sign in page for the users attempting to log in will ensure that bots won't be interacting with the services. Furthermore, adding salt and pepper values for the passwords will also add another layer of protection for the authentication system.

The foundation of cyberattacks against such vulnerability of an authentication system is based on the fact that passwords exist. NIST recommends organizations to implement the multi-factor authentication instead of using practices that involve solely on password rotations and strong password policies [11]. However, an attacker could also exploit the application if the session management is not correctly set up. For instance, by not logging out and simply closing the browser tab, the next user will still be authenticated and could potentially exploit the system.

4.3.2 Broken access control

To complement the authentication system, Remote Aero could enforce a stronger access control policy as security measures. This will result in policies that deny standard users from acting outside their intended permissions, preventing unauthorized modification of flight data or issuing flight commands which could disrupt mid-flight operations. Furthermore, enhancing access controls will also mitigate the risks from attackers who could bypass the authentication by, for instance, modifying the URL or any other inner application state such as the HTML elements with the help of third party tools, which modifies the API requests. Additionally, by strengthening the access control policy, token tempering or any other API related technology will also be significantly harder [11]. Here are the observations made towards Remote Aero's system displayed in Table 5, a security checklist by OWASP.

Table 5: Checklist for security against Broken Access Control

Security Requirements	**Check? (Y/N)**
Deny all private resources by default	✅ Y
Access control mechanisms that are reusable such as Role-Based Access Control	✅ Y
Model access control based on ownership (user should only be able to access or modify their own data unless they have specific permissions)	✅ Y
Prevention against bypassing restrictions (for instance, API should reject transactions bypassing transaction limits, ignoring manipulation in the frontend).	✅ Y
Directory for web listing disabled, protecting sensitive files ending with .git or .env	✅ Y
Logs and monitoring for access control, helpful to alert suspicious actions such as admin failing repeatedly to sign in	✅ Y
Protection against automated attacks and rate-limit APIs (protection against overloading the server with requests)	✅ Y / ❌N
Secure session & token management such as JWT tokens being short-lived and invalidated on logout. OAuth standards being followed.	✅ Y / ❌N

Currently, Remote Aero logs and monitors access control manually within the Backend, with no implementation of an automatic monitoring as noted in the interview in Section 4.2. Having an automatic anomaly detection system would increase the security system significantly by identifying and mitigating various threats such as DDoS/DoS or TMTO attacks. It will also compensate for the lack of DoS protection in the company's cloud infrastructure. Furthermore, limiting network traffic and monitoring it will also help against DoS attacks, which for instance, aims to overload the server with requests. Protection against automated attacks does exist in one way through the authentication system as it prevents an attacker from performing brute-force attacks via a backoff-handler.

Furthermore, OAuth2.0 is utilized, and JWT tokens are invalidated on logout. However, the JWT tokens are not short-lived, as they must remain active long enough to ensure that the drone-operations operate successfully as displayed in Algorithm 6, source code for access token's expiration time. As for the manipulation to the frontend, the set of parameters that can be modified is restricted. For instance, manipulating the drone's speed on the user interface does not affect its velocity, nor the battery life. Additionally, an attacker tempering with the displayed information to misguide the pilot is also mitigated, as the crucial APIs such as those which control the drone, require authentication tokens.

Weblisting is disabled and files are git-ignored, preventing users from accessing .env-files when actively attempting to access the files when entering a specific address.

```
ACCESS_TOKEN_EXPIRE =  X
REFRESH_TOKEN_EXPIRE = Y # X and Y is the amount of time before ex
```

Algorithm 6: Access token and their time before expiration

4.3.3 Injection vulnerabilities

Besides authentication and access control security, not prioritizing defenses against injection especially when commanding the drone is critical [11]. Injection attacks occur when the system fails to validate the user's input. For instance, due to this weakness, attackers could inject unauthorized malicious data into assets, commands, operations, or databases. These attacks include SQL and NoSQL injections, OS Command injection, ORM injections which extracts records that could be sensitive. However, as noted in

Subsection 4.1.3, Remote Mission Control utilizes Firebase, a NoSQL database, meaning that this database by default, mitigates injection attacks that targets SQL queries and ORM (Object Relational Mapping). Therefore, OWASP's security requirements for injection attacks do not apply in the same way as they do for SQL-based systems.

Table 6: Modified Checklist for security against Injection vulnerabilities according to OWASP top 10

Security Requirements	**Check? (Y/N)**
Utilize a secured API-design	✅ Y
Validate input, both for user and server-side validation.	✅ Y
For dynamic queries, avoid special characters misused by the interpreter	✅ Y
Restricting Query Scope - the use of LIMIT and other controls to minimize data leaks	❌N

Based on the observations in Table 6, where we check the security requirements made by OWASP, Remote Mission Control is integrating a secured API-design with an RBAC system, and authentication check once in the login page before any interaction with the web application and the APIs. Token validators are signed through JWT, safeguarding against attackers tempering with the tokens. Furthermore, Remote Mission Control has integrated a crucial session management which features token expiration after a set amount of time, see Algorithm 6. This ensures that attackers cannot exploit the system even if they manage to steal a pilot's computer. In short, injection attacks often rely on unauthorized requests on the server, meaning that the malicious users cannot easily inject malicious requests to the endpoints due to the designed architecture of the API system of Remote Mission Control.

The company validates input both for user and server-side validation. For instance, in the login page, sending numerous requests will overload the server, having a backoff-handler gives the client immediate feedback and prevents input errors, as displayed in Algorithm 3. Furthermore, according to the source code based on the pseudo code in Algorithm 7, the web application returns HTTP exceptions if an access token has expired. As for the server-side, there are few crucial variables that the user can temper with, limiting user input to only what is essential for the drone operations. In other words, the attack surface reduces and the user inputs are controlled, and the features are restricted for authenticated pilots only. However, the firebase rules set by the company need upgrading as the rules allow any authenticated user to read the data of the database regardless of their role, see Algorithm 7. The risk with this is that a user with low-privilege can read the data in the database, which poses a security threat, see Algorithm 7. Furthermore, the database does not distinguish between different user types, such as pilot or a regular user, meaning that RBAC is not implemented here.

```
Function: refresh(refresh_token from cookies):
    IF refresh_token is None:
        RETURN 401 Unauthorized ("Bearer")

    TRY to decode refresh_token using JWT_SECRET and HS256:
        EXCEPTION (on failure):
            RETURN 401 Unauthorized ("Bearer")

    EXTRACT expiration time and token type from decoded payload

    IF current time > expiration time OR token type is not "refresh":
        RETURN 403 Forbidden ("Bearer")

    GENERATE new access token for the user
    RETURN new access token in response
```

Algorithm 7: Refresh token pseudo code

Additionally, the security rules in the database are not yet being utilized as the company currently provides access via service-accounts. And as for dynamic queries vulnerabilities which the interpreter utilizes special characters to modify the database, firebase mitigates these risks since it is a NoSQL database. This means that the rules in the firebase are the most critical aspect of securing the database as firebase is relying on a structured query through SDK, which are parameterized. In other words, user inputs or other dynamic values are regarded as data and not part of the query structure, preventing SQL injection attacks due to attackers not being able to manipulate query logic through misuse of special characters.

Furthermore, the analysis on the web application indicates that it does not restrict data access in the database when called upon. For instance, once a method is called to retrieve authenticated users and their respective IDs, the entire database list will be returned as seen Algorithm 8, where the source code returns the whole list. This means that methods that fetch users, drones and their respective IDs retrieve all available data. This poses a significant risk as it might lead to potential data leaks, especially as the database grows with the growth of the company. Additionally, when fetching a large amount of data, it will also slow down the performance. Therefore, it is important to restrict the number of documents returned when fetching the information of the database. For instance, by utilizing ‘where’ clauses or other conditions which narrows the query scope, both security and performance can be enhanced.

```
def get_authenticated_users:
return db.collection(“users”).stream
```

Algorithm 8: returns a whole list of users

4.4 WebRTC & WebSocket

Remote Mission Control utilizes websocket connections throughout the system across all its API:s, encrypting communication channels using TLS, see Table 2: Glossary in Appendix A, Extended material, where each API is connected through websocket secure (WSS). For instance, Algorithm 9 displays the pseudocode distracted forom the source code where a pilot connects through a web socket. The system disables outdated variants of the TLS, TLS V.1.0 and TLS V.1.1, following recent standards and maintaining protection against common communication attacks such as eavesdropping and MitM. API endpoints are secured by using the protocol HTTPS, providing end-to-end encryption between the drone and the web application. However, to avoid potential complications and simplify the test and developing processes, Remote Mission Control utilizes simple HTTP for localhost tests. As for the WebRTC securement, DTLS (Datagram Transport Layer Security) and SRTP (Secure Real-time Transport Protocol) are being deployed to ensure real-time media and data streams protection. In summary, the communication is following security standards, but it is far from perfect as DoS attacks are still possible due to the system not having rate-limiting which limits the user requests or any automatic monitoring for anomaly traffic patterns.

```
Function: pilot_connect(websocket, drone_id)
  Define allowed_roles as [Role.PILOT]

  Accept WebSocket connection from client

  Authenticate user by calling authenticate_ws() with:
    - websocket
    - allowed_roles
    - a string containing "pilot-" + drone_id

  If authentication fails (user is None):
    - Return and close the connection

  Otherwise:
    - Get the connection object for the specified drone using get_conn(drone_id)
    - Call on_connect_pilot() on the connection object with:
      - user's username
      - WebSocket connection object
```

Algorithm 9: Pseudocode for Remote Mission Control pilot connection through web socket

4.5 Security threats

This subsection is focusing on the targeted attacks described in the Technical background, checking whether Remote Mission Control might be vulnerable towards these types of attacks.

4.5.1 XSS vulnerabilities

To check the system for XSS vulnerabilities, it is important to examine the source code utilized in building the application as well as the overall architecture [14]. XSS attacks targets primarily the

Frontend, the part of the system where a user can inject malicious scripts into the webpages, indirectly gaining access to the Backend. Therefore, the first step to detecting vulnerabilities against XSS attacks was examining the Frontend code. Upon these findings, the framework React is mostly used for the system.

React uses JSX to build user interfaces. A big advantage of JSX is that it automatically escapes any user input before showing it on the page. This means that if someone tries to enter harmful code like a <script> tag, React won't run it, it will just show it as normal text. Special characters such as '<', or '&' are converted to safe equivalents instead, automatically escaping XSS attacks.

React has a function called dangerouslySetInnerHTML that allows HTML to be inserted directly into a webpage. This can be useful in some cases, such as when displaying formatted text from another system. However, using this function turns off React's built-in protection against malicious code, like XSS attacks. If used with data from users, a hacker could inject harmful scripts that run in other users' browsers.

In this system, the function is not used, which is a good security decision because it reduces the risk of such attacks. The identification of this function is made through Visual Studio Code's search feature which lets the developer search the specific String, see Figure 3 where we utilized the text editor's search function.

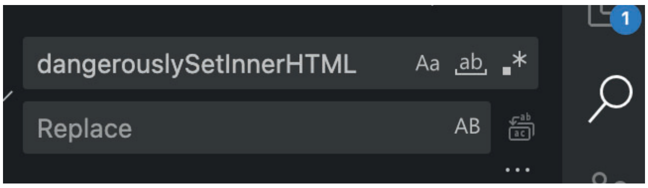

Figure 3: Visual Studio Code's search function

When data from the URL, such as query parameters, is sent to the backend or included in API requests, it should be properly validated. This is important because an attacker may attempt to inject malicious code through the URL. While React escapes data on the frontend to help prevent issues like XSS, the backend must also handle all input with care. For this reason, any data originating from the URL should be treated as untrusted by default. The login component in this case does not use any query parameters and does not use data from the url. The input is taken from input fields. Therefore, there is no XSS risk from URL-based input in this part of the application.

Additionally, Remote Mission Control is also preventing cookies from being accessed by JavaScript by setting httponly = true as seen in Algorithm 10. However, this security feature is solely focusing on protecting the cookies from being stolen by the attackers and limits the damage of an XSS attack if the web application is vulnerable against this type of attack.

Repeatedly, the most general method to protect against XSS is to validate user inputs. However, another approach is to utilize content security policy (CSP), a method to distinguish between malicious activity and legitimate user behaviour. One way of doing this is to whitelist trusted domains while blocking others which Remote Mission Control finds nontrustable. Preventing malicious script from attackers being loaded from external and untrusted sources. This method has not yet been implemented in Remote Mission Control based on the analysis of the authors of this thesis.

4.5.2 CSRF vulnerabilities

By the author's observations, React does not provide built-in CSRF protection [14]. Rather, the developers need to implement security measures against this type of attack. Subsection 4.1.2 mentions this, but Remote Mission Control does utilize JWT tokens for its authentication system.

Normally, JWT tokens help mitigate CSRF attacks due to the tokens being included in the authorization header rather than being stored in cookies, which CSRF exploits to steal data. This means that an attacker cannot force a victim's browser to send a malicious request with the required authentication that is needed to grant the attacker the authenticated session.

However, our analysis indicates that Remote Mission Control does store the JWT tokens in cookies and sends them with each request made by the user. This situation introduces a CSRF risk, as an attacker exploits the cookies which are sent automatically in each request by the browser, meaning that an attacker could target this system by tricking a logged-in user through an unintended. To combat this:

```
Code
```

```
response.set_cookie(
    key="refresh_token",
    value=refresh_token,
    httponly=True,

max_age=REFRESH_TOKEN_EXPIRE_TIME_S,
    samesite="Strict",  # Adding this
)
```

Algorithm 10: CSRF mitigations

SameSite = Strict has to be set up as displayed in Algorithm 10, which prevents cookies from being sent with cross-origin requests, resulting in a CSRF attack failing due to the malicious command not being sent as the authenticated user. Alternatively, SameSite can also be set up as SameSite = Lax, which allows cookies to be sent from certain cross-origin requests, such as top-level navigation, a situation where the user clicks on a link from for instance, exampleA.com to exampleB.com, sending lax cookies with it. Setting SameSite to Lax also protects against CSRF for requests such as POST, PUT, and DELETE, a less restricted approach but may allow CSRF attacks in scenarios where the authenticated user is tricked into a malicious link, a GET-request.

Furthermore, adding CSRF tokens adds an extra layer of security, as the tokens are tied to the user's session and ensures that the token is validated and originates from the authenticated user for each POST-/PUT-/DELETE-request, especially as the company plans to add Single Sign-On (SSO) into the system as multiple system might pose a higher security risk towards this type of attack, depending on how it is implemented. JWT tokens themselves do not detect malicious activity or attacks, they simply ensure the integrity of the data, keeping it intact and verifying that the request is authenticated.

Additionally, the absence of a strong logging and monitoring system within the web application is also a vulnerability. By implementing automated logging and real-time monitoring mechanisms, the likelihood of CSRF attacks can be significantly reduced, as detecting abnormal activities such as unusual API-requests or suspicious requests can help trigger alerts, enabling system administrators to take immediate action before any damage is realized.

Other forms of protection against CSRF includes reauthentication as a demand for critical requests. For instance, landing a drone or canceling the drone operation should require a re-authentication. Furthermore, protection for the user interface features such as confirmation pop-ups and CAPCHA is also a method to protect against CSRF attacks. Such methods are used to confirm the user's intent, ensuring that the request is from the user and not from an automatic JavaScript request generated from a bot. Additionally, Remote Mission Control users can also mitigate CSRF attacks by logging out, resulting in the termination of the session and cookie after an operation is finished.

4.6 Emerging technologies

This section explores emerging technologies which Remote Aero demands that the authors explore. Moreover, we research the related emerging technologies to the security of Remote Mission Control's system.

4.6.1 SSO Security

The emerging technology SSO provides advantages such as enhancing user experience and reducing the amount of passwords stored in the database as explained in Section 2.6. However, from the authors observations, SSO might introduce the risks of CSRF attacks due to users not needing to re-enter credentials to remain logged for extended periods. For instance, a user getting tricked by an attacker into unknowingly making an unwanted request on an active session when browsing on other websites.

The purpose of SSO is to minimize the need for multiple authentications and grant access to multiple applications or systems depending which type of SSO protocol that is utilized. This leads to an ideal scenario for a CSRF attack to take place, which exploits the SSO functionality and could potentially compromise several systems or applications at once. Furthermore, common defenses against CSRF, such as disabling SameSite cookies and utilizing metadata headers may not be as effective [12], see Section 4.5.2. These security measures are designed to ensure session binding within the workflow executed in one and single website. However, due to SSO relying on cross-site requests, such defenses would disrupt the execution of SSO and render the SSO functionality unusable.

To mitigate this risk, security protocols such as OAuth 2.0, combined with standard CSRF defenses, should be implemented. For instance, instead of setting SameSite = Strict, using SameSite = Lax will not break the SSO functionality while still providing a level of CSRF protection.

According to the interview, Section 4.2, the main purpose of this emerging technology is to add a feature where the user can authenticate with a third-party provider such as google, apple or microsoft. This enhances the complexity as the user is redirected to the provider's login page, and forwarded back to the application after having successfully signed in, see Figure 4.

This introduces additional authentication paths, and potential attack vectors as malicious actors can intercept this data flow, common attacks include MitM, especially if the web application utilizes multiple service providers which requires careful monitoring of the authentication flows, see Figure 3 which displays the dataflow for SSO. However, selecting reputable service providers such as Google generally offers benefits, as these companies are known as strong and reliable third-party organisations, which can offer reliable SSO solutions. Additionally, selecting a reputable service provider is not sufficient, in order to implement SSO securely, identity provider must also be protected to effectively mitigate vulnerabilities on both ends. Furthermore, selecting SSO can also reduce the risk of potential vulnerabilities, especially if Remote Aero has a custom-built authentication system.

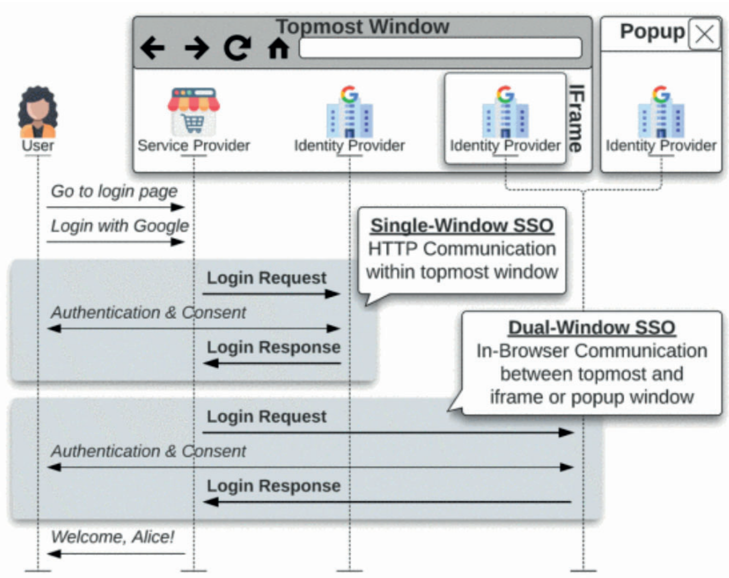

Figure 4: Dataflow for SSO. Source: Figure 2 in [12]

Chapter 5

Analysis

This chapter discusses the results and its findings on Remote Mission Control with the aim to identify the strengths and weaknesses. The purpose is to provide a complete analysis while also summarizing the recommendations made by the authors in Chapter 4.

5.1 Existing security measures

The system already has multiple robust security measures implemented. One of them is the use of RBAC, which assigns different roles to the users and gives them different access depending on their role, like pilot or spectator. This helps ensure that users can only access the parts of the system they are authorized to use, which reduces the risk of broken access control, insider threats, or misuse of sensitive API endpoints.

The system also has a backoffhandler implemented, which increases the delay after each failed attempt. After 5 failed attempts the user is temporarily blocked. This significantly reduces the risk of brute-force attacks and dictionary attacks.

Additionally, the system uses JWT for session management, signed using the HS256 algorithm, which is based on HMAC with SHA-256, a widely accepted hashing method approved by NIST. The tokens are time-limited, requiring users to re-authenticate after short intervals, which adds an extra layer of protection. This setup helps protect the system against attacks such as session hijacking and token tampering.

Furthemore, all communication between the frontend, backend and the drone is encrypted using TLS. The system also secures real-time channels and data streams through HTTPS, WebSocket Secure (WSS), and WebRTC with DTLS and SRTP. This helps prevent man-in-the-middle attacks and keeps sensitive data protected in communication.

In addition, the use of Firebase as a NoSQL database removes the risk of traditional SQL injection attacks, since no raw SQL queries are used. Finally, the frontend is developed in React, which automatically escapes user input and provides basic protection against cross-site scripting (XSS). Cookies are also marked as HttpOnly, which prevents JavaScript from accessing them.

5.2 Security weaknesses and vulnerabilities

The system does not support MFA. Right now, login is based only on a password, which means that if an attacker hacks the password, they can access the system without any extra verification. This makes the system more vulnerable to brute-force attacks, phishing, and also TMTO attacks. Adding MFA would benefit the system and make it much harder for attackers to succeed and would significantly improve the overall security of the authentication process.

Another part of the system that needs additional layer of protection is the implementation of CSRF mitigations. Although the authentication tokens (JWT) are stored in cookies with httponly set as true, preventing JavaScript access, they are still automatically included in every request sent by the browser. This means that if a user is logged in and visits a malicious site, the attacker could try to send a request on their behalf. Because the system does not use CSRF-tokens or any request validation, it has no way to verify whether a request was made intentionally by the user. This might leave the system potentially vulnerable to unwanted actions being triggered from outside sources, even with httponly protection implemented.

The system includes protection against XSS through React, which escapes user input and blocks simple script injection. However, it does not implement a CSP, which could offer an additional layer of protection. CSP allows the system to define which scripts are allowed to run, and blocks all others.

The web application can also benefit more with a better automatic threat detection and real-time monitoring system. Logging is currently done manually, and there is no system in place to detect attacks as they occur. This could make it harder to identify threats like DDoS, TMTO attacks, or unusual traffic patterns before they cause harm. This increases the risk of attacks going unnoticed. A suitable improvement would be to implement an anomaly detection system.

5.3 Trade-off between security and usability

While we have found potential vulnerabilities in Remote Mission Control, implementing all suggested security measures is not necessarily appropriate, as it introduces unnecessary complexity. This is especially true for web applications such as Remote Mission Control which is developed for sea rescue services who want to operate quickly in critical scenarios. For instance, MFA does increase the overall security of the platform but comes at the cost of reduced user experience, as it requires users to spend additional time on authentication which can be problematic, especially for pilots who quickly need access to the application in emergency situations.

Similarly, a re-authentication feature has potential downsides as this feature might hinder pilots ability to operate efficiently during critical operations such as landing. Additionally, increased complexity might also introduce costs for educating users who are less technologically experienced.

The current implementation of access tokens indicates that the tokens are not short lived. This might introduce security vulnerabilities if a malicious actor compromises and hijacks a token due to them remaining valid for an extended period. One option could be reducing the token lifespan as a countermeasure. However, overly short lived tokens may not be suitable as tokens must remain valid long enough to support pilots during critical operations. Therefore, forcing re-authentication due to token expiration might not be the best idea as it could disrupt the mission severely.

Furthermore, as noted in Subsection 4.5.2, setting the attribute sameSite to strict might not be a suitable option, although it provides stronger protection against CSRF attacks. This configuration might be overly restrictive for pilots who require extra flexibility across different systems and tabs. Therefore, as Section 4.6.1 explains, configuring sameSite to lax might bring a better balance between usability and security.

Finally, while MFA has clear benefits, the implementation of the SSO functionality might be overall better suited here than MFA. SSO creates lesser constraints on the database and simplifies the login process, meaning that this technology is far more suited for drone application in mission-critical environments than MFA.

5.4 Future improvements

To improve the security of the system, there are several measures that could be implemented. First, adding MFA would make it harder for attackers to access the system with only a cracked password.. Stronger password rules and a CAPTCHA on the login page could also help prevent brute-force attacks. To reduce the risk of CSRF, the system could use CSRF tokens, and a Content Security Policy should be added as an extra layer of protection to strengthen the system against XSS. It is also recommended to introduce real-time monitoring or anomaly detection to identify malicious activity early.

Single Sign-On has not yet been implemented, but it is planned as a future feature. While SSO can improve the login experience and reduce the need for multiple passwords, it may also introduce new risks.

For example, users stay logged in longer, which increases the chance of CSRF attacks if a malicious site triggers unwanted requests. To secure SSO properly, the system needs to include CSRF protections and it is also important to choose a trusted *identity provider*, a system entity that provides user authentication services for storing and verifying user identity.

Chapter 6

Conclusion

This section concludes the key findings and insights derived from addressing RQ 1 to RQ 4 posed in this study. Furthermore, by examining the key findings of various aspects of the architecture, this conclusion can determine valuable reflections on future work regarding technical enhancement and implementations. Additionally, the ethical reflection is also going to be discussed to reflect on the direction that this thesis has gone through.

- RQ 1: What security measures exist in Remote Mission Control's system and how effective are they against cyberthreats?

As discussed in Chapters 4 and 5. The most essential security measure implemented is the custom-based authentication system, serving as the first line of defense against the attacks threatening the web application. The security feature offers a variety of security implementations which are mostly effective against the cyberthreats discussed in this thesis. For instance, such a threat includes attacks on the authentication system, including brute force and dictionary attacks due to the back-off handler. Furthermore, the authentication also follows recommendations by the NIST such as utilizing the standard SHA-256 hashing algorithm for storing the passwords. The web application also features a RBAC system, a method of limiting the access control to specific users, lowering the risks of insider attacks as well as even if an attacker successfully gets access to an authenticated user, the attacker cannot cause damage to the system unless the stolen user credentials are high-privileged accounts. Additionally, JWT tokens are deployed, effectively protecting the tokens from being tampered with, the user also has to reauthenticate after some amount of time that has passed to ensure the legitimacy of the user, as displayed in the pseudocode in Algorithm 6.

- RQ 2: What are the primary cyberthreats that target the web system and what vulnerabilities exist in Remote Mission Control's system?

The primary cyberthreats focused in this thesis are MitM, TMTO, SQL Injection, XSS, CSRF, and DoS. Most of these attacks aim at stealing the cookies, but from the analysis results, the web application has protections against MitM, TMTO and SQL injection attacks. However, the system can add extra protection against XSS, CSRF and particularly DoS attacks. The web application does possess some protection against CSRF and XSS but from the authors observations, Remote Mission Control benefits from enhancing the system security further and to implement suggested improvements written in Chapter 4. For a list of potential vulnerabilities, see Table 1 in Appendix A, Extended material.

- RQ 3: What countermeasures can be implemented to mitigate these vulnerabilities that exist in this web-based drone platform? How can we enhance the system against the vulnerabilities?

By following the OWASP standards for web application security, the potential vulnerabilities found in this analysis will decrease. Implementations such as MFA, password rotation, adding salt and pepper may enhance the authentication system significantly, especially against TMTO attacks. As for the DoS attacks, regulating traffic and implementing a strong logging and monitoring system that can detect abnormal activities on traffic will greatly reduce the risk of a DoS occurring, which goes beyond the basic logging system which is provided by Firebase. Additionally, adding a CAPCHA and tracking system which monitors the cross-site requests, and limiting what requests can be sent, the likelihood of CSRF and XSS attacks will reduce.

- RQ 4: What are the security implications of implementing Single Sign-On and how does it impact user experience?

The implementation of SSO will enhance the user experience and impact positively on Remote Mission Control's database significantly. By allowing users to authenticate via third-service providers such as Google or Apple, the system minimizes the need for storing and verifying credentials. However, this feature requires a secured implementation of the SSO functionality as an improper SSO system could expose Remote Mission Control platform to threats such as MitM attacks.

Future work

Remote Mission Control will transition from Firebase, a NoSQL-database to supabase, a SQL-based database, which may introduce the risk of SQL injection attacks as a potential attack vector, an issue which was previously considered irrelevant in Chapter 4. To combat this, the developers should correctly utilize parameterized queries to generate SQL-formated code and avoid the use of "EXECUTE" with raw

SQL queries. This prevents the attackers from altering the query behavior, meaning that SQL injection attacks will not be possible. However, these observations will be saved for future work.

Additionally, future research could be dedicated to secure the database. One approach is to encrypt the database and implement encrypted indexes and a trusted handler which works as a middle-man between the database and clients to manage the encrypted requests and responses. This approach would make the query processing more secure, and lower the risk of plaintext data getting exposed.

Furthermore, future work can consider utilizing an enhanced backoff-handler while also providing a DoS protection on the authentication modul to prevent attackers from potentially locking the user out. For instance, instead of suspending the user after five failed attempts, one approach would be to temporarily soft-lock the application and force the user to be redirected into a MFA as a verification that the user is legitimate. To defend against automatic brute force attacks, the system could integrate a CAPCHA to detect scripts or bot activities.

Regarding potential DoS attacks vulnerabilities, the implementation of essential security features such as an anomaly detection system could significantly mitigate DoS attacks, as an effective detection system could be capable of identifying abnormal traffic activities in real-time and thus prevent the malicious act before disruption could occur. Furthermore, a proper logging and monitoring system could complement the anomaly detection system, flagging for unusual activity or on-going attacks. These combinations might enhance Remote Mission Controls resilience and give the administrators correct information and time to rapidly respond against such threats.

X509-certificates will be added to enhance the security of Remote Mission Control. However, the certificate's effectiveness is heavily dependent on implementation and its management. Misconfigurations such as failures in the certificate chain, weakness in the encryption algorithms such as using outdated algorithms or mishandling private keys can increase the risks of malicious actors succeeding in gaining access to the web application. Therefore, further analysis needs to be done, but these analytics are beyond the scope of this thesis. However, from a quick overview, the authors found no weaknesses in the encryption algorithms.

Additionally, this thesis has only included a few web-based cyber attacks, and there are potentially a lot more attacks that need to be addressed. For instance, GPS-spoofing is a significant threat that could interfere with remote drone operations carried out by the sea rescue services. This attack usually happens when the attackers attempt to manipulate navigation signals, resulting in severe consequences if left unmitigated. Therefore, due to this type of threat, further research has to be made. Furthermore, compliance with relevant security standards, such as NIS2 and guidelines for IoT is critical for protecting sensitive data and needs to be addressed in future studies. Third party organisations and their systems that are collaborating with Remote Aero must also be taken into consideration when it comes to future cybersecurity analysis. Also, SSO needs to be implemented as it will enhance the user experience greatly.

Ethical reflection

Reflecting from an ethical standpoint, a Non-Disclosure Agreement (NDA) has been signed with the company. This was to ensure that no classified or sensitive information has been disclosed in this study. This includes any potential vulnerability which can harm the company's reputation and its products.

Furthermore, prior to the publication, an approval from the company was required in the event of disclosing sensitive information, and if any issues arise, the topic was then addressed with the company. For instance, we provided an early draft of this document to allow sufficient time for Remote Aero to fix the potential vulnerabilities documented in this report. This way, we could assure that neither party was dissatisfied with the project. Additionally, no ethical penetration testing or ethical hacking was conducted on the system, but rather an analysis was carried out through the source code.

As mentioned in Section 1.5, in the development of alternative methods such as utilizing drones remotely in critical operations, providing the highest level of protection is necessary, especially when human lives are at stake. This involves implementing strong security measures to protect the system controlling drones remotely from malicious actors who threaten the system. Therefore, an analysis of the cybersecurity of Remote Mission Control is a must.

By having a properly secured web system, trust and reliability will be built with the company's customers and partners. This would result in continuous development of Remote Mission Control. Leading to the future deployment of this drone application in real operations to effectively support sea rescue services.

References

[1] T. U. Daim and M. Dabić, “*Cybersecurity, A Technology Landscale Analysis*”. Switzerland, Springer International Publishing, ISBN: 978-3-031-34843-3, 2023

[2] I. Priyadarshini, C. Cotton, “*Cybersecurity Ethics, Legal, Risks, And Policies*”. USA, Apple Academic Press, ISBN: 978-3-031-34842-6, 2022

[3] K. Rajamohan, Book: “Internet Of Drones”, Chapter: “Review of Medical Drones in Healthcare Applications”. USA, New York, CRC Press, pp 16, eBook: ISBN9781003252085, 2023

[4] M. Jazayeri, "*Some Trends in Web Application Development*" Future of Software Engineering (FOSE '07), Minneapolis, MN, USA, 2007, pp. 199-213

[5] A. Blau, A. Burt, B. Groysberg, Roman V. Yampolskiy, “*Cybersecurity : The Insights You Need From Harvard Business Review*”, Boston, Massachusetts, Harvard, ISBN: 978-1-633-69787-4, 2019

[6] D. Stuttard and M. Pinto, “The Web Application Hacker’s Handbook: Finding and Exploiting Security Flaws”. Indianapolis, Indiana, Wiley Publishing, ISBN: 978-1-118-02647-2, 2011

[7] P. Yaworski, “Real-World Bug Hunting: A Field Guide to Web Hacking”. San Francisco, No Starch Press, ISBN: 978-1-59327-861-8, 2019

[8] J. M. Biju, N. Gopal, A. J. Prakash, “CYBER ATTACKS AND ITS DIFFERENT TYPES”, International Research Journal of Engineering and Technology (IRJET), Volume 6, pp. 4849-4852, e-ISSN: 2395-0056, Mar 2019

[9] C. J. Brooks, C. Grow, P. Starks, and D. Morgenstern, “Cybersecurity Essentials”. Indianapolis, Indiana, Wiley, ISBN: 978-1-119-36069-3, 2018

[10] W. Stallings and L. Brown, Computer Security: Principles and Practice, 4th ed., Global ed. New York, NY, USA: Pearson Education Limited, 2018, ISBN: 978-1-292-22061-1. [Accessed: Feb. 6, 2025]

[11] OWASP, "Authentication Cheat Sheet," OWASP, 2021. [Online]. Available: https://cheatsheetseries.owasp.org/cheatsheets/Authentication_Cheat_Sheet.html. [Accessed: Feb. 6, 2025]

[12] L. Jannett, M. Westers, T. Wich, C. Mainka, A. Mayer and V. Mladenov, "*SoK: SSO-MONITOR - The Current State and Future Research Directions in Single Sign-on Security Measurements*", 2024 IEEE 9th European Symposium on Security and Privacy (EuroS&P), Vienna, Austria, 2024, pp. 173-192

[13] [1] J. De Clercq, "*Single Sign-On Architectures*" in **Infrastructure Security**, G. Davida, Y. Frankel, and O. Rees, Eds., Berlin, Heidelberg: Springer, 2002, vol. 2437, pp. 37–47, doi: 10.1007/3-540-45831-X_4.

[14] T. Gamma, N. Gerasimenko, “React Security Labs”, BA Documentation, OST Eastern Switzerland University of Applied Sciences, School of Computer Science, Version:1.0, 2024-06-13 22:12:23

Appendix A: Extended Material

This section is dedicated to showing extra material for a deeper understanding of this thesis.

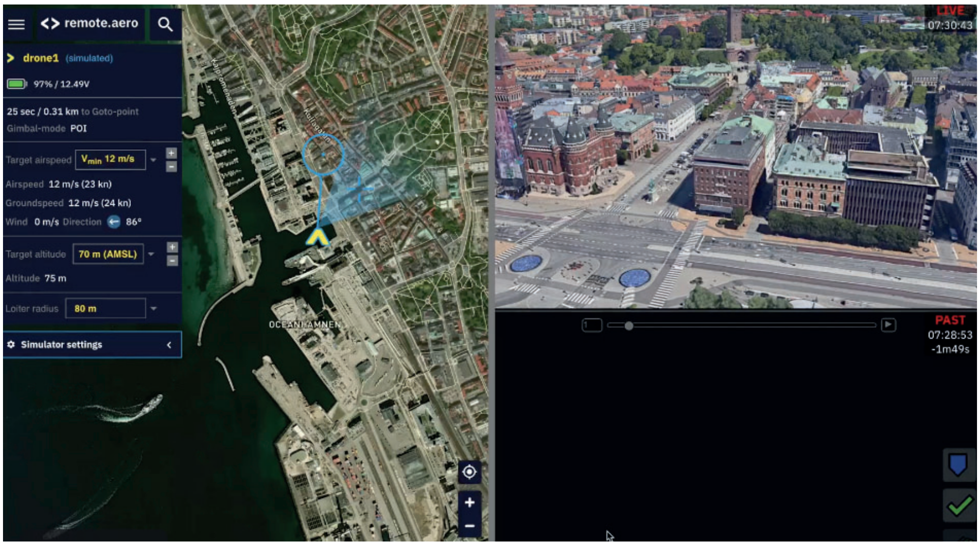

Figure 1: Remote Mission Control Simulation mode

Table 1: Short summary of findings of vulnerabilities, for further details check the Section/Figure/Solution description.

Potential vulnerability	**Section/Figue/Solution**
1. Lack of anomaly detector / automatic detection of cyberthreats	Needs to be implemented, see Table 4/Section 4.3.2
2. Cloud operator does not provide DoS-protection	Solution: vulnerability 1
3. Replacement for backoff-handler	Solution: reduce likelihood of DoS and reimplement backoff-handler
4. Strengthen passwords a. Lack of salt and peppar	Section 4.3.1

b. Allowing weak passwords such as '1234'	
4. No MFA or two-factor authentication	Section 4.3.1
6. No CAPCHA implemented	Section 4.3.1
7. JWT tokens not shortlived	The fix of this vulnerability is optional as tokens must remain active long enough, see Section 4.3.2
8. Not restricting Query Scope	Section 4.3.3
9. Not whitelisting/blocking malicious activity, XSS vulnerability	Section 4.5.1
10. Storing JWT tokens in cookies and sending them, CSRF risk	Section 4.5.2
11. Not requiring re-authentication for critical drone operations such as canceling drone operations/landing, CSRF risk	Section 4.5.2
Potential problems in the future	
12. Potential SSO vulnerabilities as it collides with CSRF solutions	Section 4.6.1

Table 2: Glossary

Glossary	**Section/Figue/Solution**
ADS	ADS stands for *Automatic Dependent Surveillance* and is an aviation system where aircraft automatically send data such as their position to ground stations or other aircraft. It improves situational awareness and improve safety by keeping track of where planes are.
AIS	AIS stands for *Automatic Identification System* and is a maritime safety system used for tracking ship data. This data includes ship positions, identification and their course using GPS and radio signals. In Remote Mission Control, it is integrated into the system to enhance the awareness of the surrounding vessels.
Websocket	A communication protocol, useful for real-time data which is used for the

	drone telemetry or video feed in Remote Mission Control
HTTP	HTTP stands for *Hypertext Transfer Protocol*, and is a protocol used mainly for communication between web browsers and servers. A more secured version is the HTTPS which stands for *Hypertext Transfer Protocol Secure.*
WebRTC	WebRTC stands for *Web Real-Time Communication*, and is a set of protocols and APIs that let browsers share audio, video, and data directly with each other in real time.
SRTP	SRTP stands for *Secure Real-time Transport Protocol*. It secures audio and video streams, making sure data is safe during transmission making it ideal for drone operations.
DTLS	DTLS, stands for *Datagram Transport Layer Security* and is a variation of TLS designed to provide security for fast and connectionless data. This makes it design-wise ideal for unreliable transport layers like UDP.
Rate Limiting	Its purpose is to help control the number of requests a user can make to a system in a given time period. This helps prevent abuse and DoS/DDoS attacks
OAuth 2.0	A system that lets websites or apps access limited user data without needing the user's password leading to the user info being kept safer.
TLS	TLS, stands for *Transport Layer Security* and is cryptographic protocol that ensures privacy and data integrity between two communicating applications over the internet.

Printed by Tryckeriet i E-huset, Lund 2025

Series of Bachelor`s theses
Department of Electrical and Information Technology
LU/LTH-EIT 2025-1055
http://www.eit.lth.se