Institute of Informatics and Mechatronics

Bachelor's Thesis

in the field of study of Computer Engineering and Mechatronics

Cloud Managed Unmanned Aerial System

Bruno Axel Kamere
Student's transcript number **030756**

Seminar conducted by
Szychta Elżbieta, prof. dr hab. inż.
Thesis supervisor
Ocetkiewicz Tomasz, mgr inż.

Bydgoszcz, Poland 2022

This work is dedicated to my loving and very supportive parents . . .

Acknowledgements

Looking back at my journey that led to this point, I would like to take some time and thank everyone that has helped me throughout my studies, and design, development, and testing of this final engineering work. The moral support received throughout this period is unprecedented.

I would like to first and foremost thank my lovely parents. They have been there for me in each single step that led to where I am and what I have achieved today. They have encouraged me, believed in me, and supported me in various ways. Thank you.

Next, I would like to also thank a lot my thesis supervisor, Ocetkiewicz Tomasz, mgr inż.; first, for believing in my ability to conduct my research towards the execution of my project and second, for the unprecedented advice and support I have received throughout my studies and throughout my final engineering research and project execution. Thank you.

I want to also take a moment to thank the WSG university's institute of informatics and mechatronics professors and staff for the knowledge and skills they have provided me throughout my studies. They have inspired me to like more my field of studies, due to the challenges provided to me along the way. Thank you.

Finally, I would not finish without thanking my classmates, colleagues and friends who have been part of my support system and have given me ideas and support during throughout my studies, until this day. A big thanks to you.

Cloud Managed Unmanned Aerial System

Abstract. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

Keywords: AWS, UAV, UAS

DECLARATION

I, the undersigned ....,
student of the University of Economy in Bydgoszcz, declare that the following Bachelor's/Engineering/ Master's/ Thesis* entitled ....
....
....
has been written on my own. It means that:

1) during writing the thesis I have not used any help from other people (apart from necessary the consultations conducted within the didactic process), mainly I have not commissioned writing the thesis or it's part to other people and I have not rewritten it or its part from other people;
2) After the use of the literature and source materials I have mentioned the author, the title and the source according to the quotation rules.
3) The thesis and its fragments has not been used as the basis of other procedures aiming at conferring degrees and academic and professional titles;

I declare that I am aware that if my thesis infringes rights connected with copyright and diplomatic regulations, it may be invalidated by the University even after the degree examination.

I also declare that the media attached and the version in the Electronic System ISAPS contains all files connected with the following thesis, which have been used in a printed version.

I prove he authenticity of the above declaration with my handwrite signature

Bydgoszcz,......................... y. ....
(signature)

DECLARATION OF AUTHOR'S WILL

Surname and name of the author..
The title of the thesis ....
....
Address of Residence ....

I agree to make above mentioned thesis of my authorship available for scientific and didactic purposes with the consent of the Supervisor. The form of sharing includes: sharing of a printed copy and sharing of its electronic version.

Bydgoszcz, ... y. ....
(signature)

*delete as appropriate

Contents

List of Figures

List of Tables

List of Source Codes

Nomenclature

Acronyms / Abbreviations

ALB	Application Load Balancer
AWS	Amazon Web Services
AZ	Availability Zone
CDK	Cloud Development Kit
DJI	Da-Jiang Innovations
EC2	Elastic Cloud Compute
ECS	Elastic Container Service
GCS	Ground Control Station
HHLD	High-High-Level Design
HLD	High-Level Design
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secured
IaaS	Infrastructure as a Service
IaC	Infrastructure as Code
LTE	Long Term Evoluton (Telecommunication)
ML	Machine Learning
NACL	Network Access Control List
NAT	Network Address Translation
PaaS	Platform as a Service

RDS Relational Database Service

TCP Transmisison Control Protocol

UAS Unmanned Aerial System

UAV Unmanned Aerial Vehicle

VPC Virtual Private Cloud

Chapter 1

Introduction

Unmanned Aerial Vehicles also known as UAVs or Drones, although hardly a new technology, with the first used UAV recorded in history dating back to 1849 [1], have recently gained a lot of attention from various sectors ranging from entertainment to military. This is going to have an impact that cannot be overseen over the coming years as more and more people find uses of UAVs in various applications. UAVs were initially developed to be used for military operations, mainly surveillance, but they were later armed to also enable them to perform long-distance military operations without putting humans at risk. The United States of America has used these types of UAVs mainly in the wars in the Middle East, where UAVs like the General Atomics MQ-9 Reaper also known as Predator B and Northrop Grumman RQ-4 Global Hawk have been widely deployed [2].

Despite their use in the military sector, UAVs have also been employed in other sectors such as commercial and entertainment sectors, where UAVs are being used in things like land geography mapping, industrial surveillance, photography and many more. Companies like SZ DJI Technology Co., Ltd. or Shenzhen DJI Sciences and Technologies Ltd. in full, more popularly known as its trade name DJI have had a lot of success in this area, where as of March 2021 DJI was coveringitself covers (research on the percentage of drones that DJI makes and are on the market). UAVs have also seen great use in the healthcare sector, where companies like Zipline [3] are implementing an end-to-end supply chain system that employs UAVs to supply and deliver medical supplies to hospitals in rural areas in Rwanda that are hard to reach or inefficient to reach by other means of delivery.

Rwanda has also seen great use of UAVs during the COVID-19 pandemic where UAVs were widely used by the Rwanda's Ministry of Health and the Rwanda National Police to spread COVID-19 awareness in Kigali communities [4].

As UAVs gain the market, the need to have robust UAV systems also known as UASs becomes eminent. Therefore, in this thesis, focus was put in designing and building a robust, scalable, highly available cloud deployed Unmanned Aerial System, that can easily be integrated with cloud services like Amazon Web Services also known as AWS to provide a solution where UAV pilots can control UAVs from virtually anywhere in the world. The proposed system comprises of a UAV flying with onboard compute that has an LTE datalink to a ground control system also known as GCS, dashboards and a command-and-control center application running in a highly available and fault tolerant AWS cloud infrastructure. The focus of this thesis is to therefore assess the possibilities of implementing such a solution in an efficient, resilient, reliable, and highly available manner and discuss on the pros and cons of the solution.

The proposed solution, as seen in the high high level design in figure 1.0.1, was developed following the best industry standards in software development and architecture as is going to be described in detail in the next chapters. This thesis is also going to discuss the developments that have already been made in this area as well as areas that need further research and development.

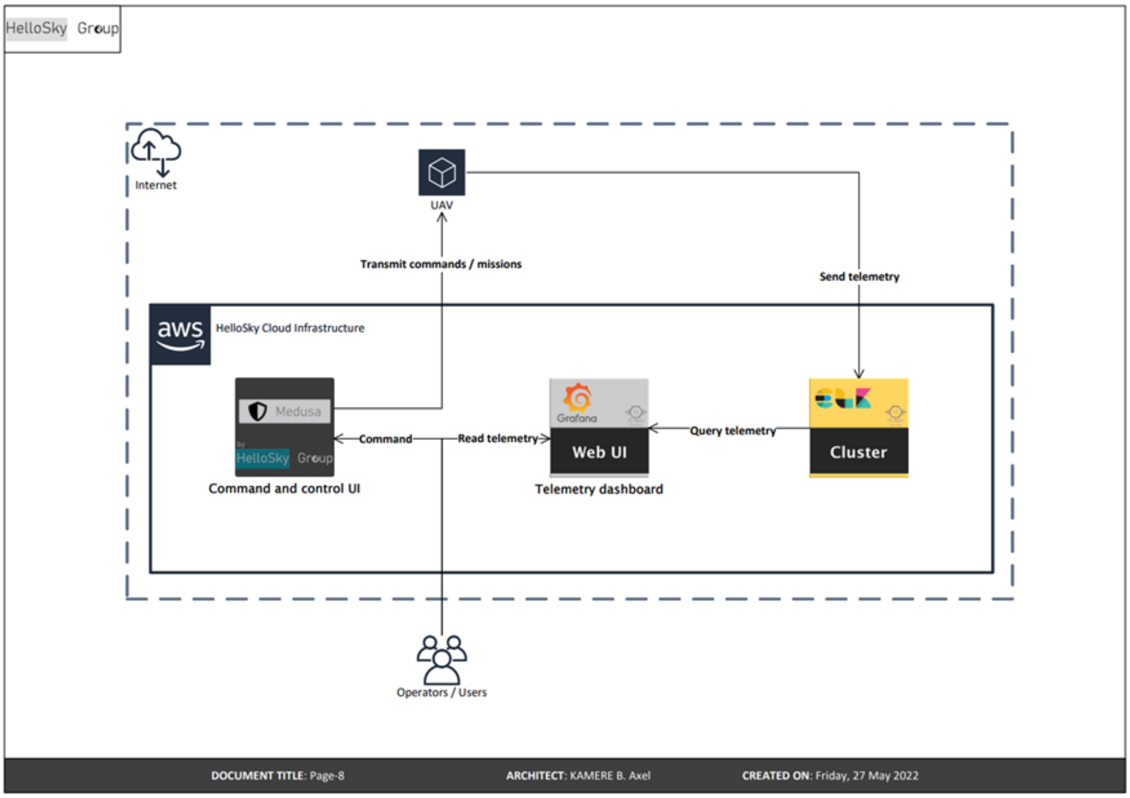

Figure 1.0.1. Proposed system high-high-level design.

Source: Own creation. Designed with Microsoft Visio. Refer to 2.4.6.

1.1 Related work

UAVs and UASs in general is a field that has undergone substantial development through various researches done by scientists, engineers and academicians.

One of the challenges still faced by UAVs, especially in the capability of being able to deploy them in urban areas, is the safety of their operations. Being able to build a

UAV with highly effective collision avoidance algorithms is a still a field under active research. And this is one of the main challenges that need to be solved for the world to see robust autonomous UAVs employed widely in communities for various use cases. Pedro et. al have studied on how UAVs can be made more resiliant and safe with the help of artificial intelligence, machine learning and the likes. In their article "Framework for fully autonomous UAVs" [5], they reviewed the current collision avoidance algorithms for both static and dynamic objects and proposed a conceptual framework to improve more the safety and realiability of UAVs.

<ADD ONE MORE USECASE??>

1.2 Use case

As UAVs emerge, there will be a need to be able to centrally manage a fleet of UAVs. Depending on the UAV use case, operators might need to also control them at a long distance beyond eyesight. A UAV operates as part of a system comprised of multiple other components that support the operation of a UAV. The main components are a Ground Control System, (Research on the main components of a UAV). UAVs can either be fully autonomous, fully manual, or semi-autonomous. UAVs can also be employed in various use cases, below are various scenarios in which UAVs can be used (CITE SOURCES OF THE BELOW USE CASES)

- Agricuture.
- Facility inspection.
- Terrain mapping.
- Shipping and delivery.
- Search and rescue.
- Law enforcement.
- Military reconnaissance / Surveillance.

For a UAV to perform any of the above, it needs to meet certain criteria, a UAV should:

- Have onboard computer to process mission commands on the fly.
- Have onboard key components like,
  - Sensors, depending on the mission.
  - Cameras.
  - Battery.
  - LTE modules or Satnav modules to allow communication with ground control.

- Have LTE or Satellite communication to enable the UAV to set up a datalink with the ground control. The UAV would have to send data such as
  - Ground speed.
  - Altitude.
  - Battery levels.
  - Yaw.
  - Location.
  - Direction.
  - Sensor data.
  - Send the data frequently for real-time or near real-time communication.
  - Be able to react and if necessary, take evasive maneuvers when:
    * On collision course.
    * The batteries are low on power.
    * Out of connectivity range.

1.3 Problem definition

1.4 About HelloSky group

Across this thesis, there will be mentions of the name "HelloSky group". Several designs built for the project as well as source codes all have mentions of HelloSky group or hsg in abbreviations.

HelloSky group is a company name that I came up with to brand my work done and future developments that will be made on this project and many other related projects that will be built in the future. HelloSky group in itself was though as a group company that will have multiple child companies, and in the scope of this project, it will be used to represent the part of the company that is envisioned to deal with aerial monility, hence being the scope of the thesis. Figure 1.4.1 shows the HelloSky group logos used throughout the thesis project.

(a) Colored 500 x 200.

(b) Black and white 500 x 200.

Figure 1.4.1. HelloSky group logos.

Source: Own creation. Designed with Affinity Designer. Refer to 2.4.4.

Chapter 2

Theory

In this chapter, key background concepts and methodologies used in the thesis are going to be discussed. The chapter is going to discuss explain what is meant by unmanned aerial system and its components.

The chapter is also going to discuss on the cloud provider, Amazon Web Services (AWS), used to host various components of developed system, simulation and software development tools used, as well as laws and regulations around unmanned aerial systems.

2.1 Unmanned Aerial System

bla bla

2.1.1 Unmanned Aerial Vehicle

bla bla

Classification of Unmanned Aerial Vehicles

bla bla

2.2 Amazon Web Services

Amazon Web Services, commonly known as AWS, is a cloud platform provided by Amazon that provides various service offerings such as platform as a service, PaaS, and infrastructure as a service, IaaS[6]. AWS makes it easy for developers, engineers and businesses to deploy scalable, resilient, agile and highly available infrastructures for databases, servers, applications, storage, analytics, *et cetera*. AWS offers attractive and cost saving payment strategies of which there are pay-as-you-go, save when you commit, and pay less by using more[7].

Cloud computing is an emerging technology that has revolutionized how businesses go online. Cloud computing has been and still is of great use in various industries, including the aerospace and energy industries. Burak et al developed a cloud and edge solution running on AWS that aimed at increasing turbine maintanance inspections' efficiency through automation and a serverless AWS architecture while reducing operations cost[8]. A serverless architecture is a type of architecture where servers' configuration and patching is taken care by the provider, thus allowing developers and engineers to focus on the actual resources, applications, databases *et cetera*, to be deployed. The solution proposed by Burak et al was comprised of drones, machine learning and Internet of Things running on cloud and edge.

The proposed solution in this thesis also takes advantage of what AWS and clound computing offers. Several components, like the ground control system, of the proposed solution are running on AWS. See the high-high-level design in figure 1.0.1.

2.2.1 Infrastructure as code

Infrastructure as Code also known as IaC, a technique very often used in the DevOps and automation community, is an infrastructure that is provisioned through code and scripts written in familiar programming languages like Python, PHP, Node.JS, C# *et cetera*. The infrastructure deployed through code can be servers, databases, firewalls, data centers *et cetera*. The main advantages of defining an infrastructure as code are:

- Improved efficiency and consistency.
- Reduced human error.
- Infrastructure agility. An infrastructure defined as code can be deployed as many times as needed, which reduces the effort invested by developers in case a replica of an environment is needed elsewhere.
- It allows developers to take advantage of programming languages features like loops, variables *et cetera* to build more agile infrastructures.
- The infrastructure can be versioned and tightly controlled. Since the infrastructure is basically standard code, it can be versioned with various versioning tools like Git or Subversion. This facilitates maintanance and makes the infrastructure easy to be rolled back, in case of issues.
- It helps with cost savings. Since the whole infrastructure is basically deployed automatically through code, engineers can then shift their focus to work on other important tasks.

In this thesis, Infrastructure as Code is used to its outmost potential. The AWS infrastructure is deployed as code using the AWS proprietary software development framework called AWS Cloud Development Kit or AWS CDK. AWS CDK is an open source kit provided by AWS that allows engineers to define IT infrastructures on AWS using familiar programming languages. In the source code 1 is an example snippet from the AWS CDK app developed for the proposed solution in this thesis. The snippet represents a part that adds DNS records to the AWS Route 53 service using standard Python code.

```python
#!/usr/bin/env


# Import needed libraries
import aws_cdk as cdk
from aws_cdk import (
    aws_route53 as route53,
    aws_certificatemanager as certificate_manager,
    aws_route53_targets as targets,
    Stack
)
from constructs import Construct


class Route53RecordsStack(Stack):
    def __init__(self, scope: Construct, construct_id: str, props: dict, internet_facing_alb, hosted_zone,
                 **kwargs) -> None:
        super().__init__(scope, construct_id, **kwargs)

        # TODO: #61 Apply removal policy of the hosted zone.

        # Create A record 'helloskygroup.com' pointing to the internet facing ALB alias.
        route53.ARecord(self,
                        f"{props['company_abbreviation']}-medusa-{props['environment']}-alias-a-record",
                        target=route53.RecordTarget(
                            alias_target=targets.LoadBalancerTarget(internet_facing_alb)),
                        zone=hosted_zone,
                        comment="A record for root helloskygroup.com pointing to the internet facing ALB",
                        ttl=cdk.Duration.hours(2)
                        )

        # Create A record 'www.helloskygroup.com' pointing to the internet facing ALB alias.
        route53.ARecord(self,
                        f"{props['company_abbreviation']}-medusa-www-{props['environment']}-alias-a-record",
```

```
                        target=route53.RecordTarget(
                            alias_target=targets.LoadBalancerTarget(internet_facing_alb)),
                        zone=hosted_zone,
                        record_name="www",
                        comment="A record for www pointing to the internet facing ALB",
                        ttl=cdk.Duration.hours(2)
                        )

        # Create A record 'dashboard.helloskygroup.com' pointing to the internet facing ALB alias.
        route53.ARecord(self,
                        f"{props['company_abbreviation']}-grafana-{props['environment']}-alias-a-record",
                        target=route53.RecordTarget(
                            alias_target=targets.LoadBalancerTarget(internet_facing_alb)),
                        zone=hosted_zone,
                        record_name="dashboard",
                        comment="A record for Grafana (dashboard.helloskygroup.com) pointing to the internet "
                                "facing ALB",
                        ttl=cdk.Duration.hours(2)
                        )

        # Create A record 'logs.helloskygroup.com' pointing to the internet facing ALB alias.
        route53.ARecord(self,
                        f"{props['company_abbreviation']}-kibana-{props['environment']}-alias-a-record",
                        target=route53.RecordTarget(
                            alias_target=targets.LoadBalancerTarget(internet_facing_alb)),
                        zone=hosted_zone,
                        record_name="logs",
                        comment="A record for Kibana (logs.helloskygroup.com) pointing to the internet facing ALB",
                        ttl=cdk.Duration.hours(2)
                        )

        # Create A record 'api.helloskygroup.com' pointing to the internet facing ALB alias.
        route53.ARecord(self,
                        f"{props['company_abbreviation']}-node-red-{props['environment']}-alias-a-record",
```

```
                        target=route53.RecordTarget(
                            alias_target=targets.LoadBalancerTarget(internet_facing_alb)),
                        zone=hosted_zone,
                        record_name="api",
                        comment="A record for Node-Red (api.helloskygroup.com) pointing to the internet facing ALB",
                        ttl=cdk.Duration.hours(2)
                        )
```

Listing 1. helloskygroup.com AWS CDK Python Route 53 snippet.

2.3 Simulation

bla bla

2.3.1 Webots or Ardupilot?

bla bla

2.4 Graphics and software development

The solution proposed in this thesis was built using various software development, and design tools. The choice of tools is really key to an organised and well managed project development, therefore it was important to choose the right tools for the right tools to help get the expected outcome from them.

In the next subsections, various tools during the solution development are going to be listed and discussed.

2.4.1 Microsoft Visual studio code

Visual studio code or VS code is a source code editor provided by Microsoft. It was used in this thesis as a code editor to write various codes and scripts. Microsoft VS code free license used allows users to use it for personal and commercial use.<CITE LICENSE>.

2.4.2 PyCharm by JetBrains

The AWS infrastructure was built as code using Python programming language. Due to its robustness and great Python support, Pycharm by JetBrains integrated development devolpment or IDE was the go to choice.

2.4.3 PhpStorm by JetBrains

PhpStorm is an integrated development environment created by JetBrains specifically for PHP programming language. The web interface from which the proposed Unmanned Aerial System is controlled from is built with PHP's Laravel framework, and PHP storm is perfectly optimised for development of PHP software.

2.4.4 Affinity Designer

Several UI components used across the project like icons and logos, like in figure 1.4.1 for example, were designed using Affinity Designer. Affinity Designer is a graphics tool used to design and create logos, icons, concept arts, UI designs *et cetera*.

2.4.5 GitHub

One of the fundamentals of software development and coding projects generally is to version the code so that changes can be tracked overtime. Making sure that a project is versioned and maintained centrally in a repository is very important, especially where teams are working together on a similar project. Git, one of the softwares used for code versioning, was used in this project to track changes across various components of the overall project. In fact this thesis document itself is versioned using Git <ADD LINK TO THIS THESIS ON GITHUB>, alongside other components of the proposed solution. Github then comes into play to act as the single point of truth where multiple Git repositories can be pushed and managed from. <ADD IMAGE SHOWING A TYPICAL BASIC GIT - GITHUB FLOW>

2.4.6 Microsoft Visio

Microsoft Visio is an application part of the Microsoft office family that is used for digramming and graphics visualization. It is used to build architecture diagrams and many more. In this project, Microsoft Visio was used to draw and create architecture design diagrams of the proposed solution. <ADD REFERENCE TO AN HLD>

2.5 Law and regulation

The aerospace industry is one of the highly regulated industries. And this is so for a reason, mainly safety and airspace security. <FINISH SECTION>.

Chapter 3

Methodology

In this chapter, the solution is going to be explained in details. The reason behind various design choices is going to be explained elaborately as well as the technical aspects of the project.

3.1 Approach

3.2 Solution description

3.3 Network access and security

One of the challenges with implementing a networked system, especially on cloud platforms like AWS, is ensuring that traffic flows in the expected way with proper security in place. The proposed solution, being a networked solution involving communications to and from various applications, has a rigorous network design. Figure <REFERENCE TO THE NETWORK DESIGN IMAGE> shows how network within AWS.

AWS has a concept of Virtual Private Cloud also known as VPC, which is simply an isolated private network that can be broken down into various subnets depending on the architecture. The proposed solution has 3 subnets, public, private and isolated-private subnets for each availability zone

3.3.1 Public subnet

The public subnet in this proposed solution does not contain any resources, except a Network Address Translation or NAT gateway that is used by resources in the private subnet to access the internet. Table 3.3.1 and 3.3.2 show the inbound and outbound traffic rules respectively configured on the public subnet network access control list or NACL.

Inbound traffic					
Rule	Type	Protocol	Port range	Source	Allow/Deny
100	HTTP (80)	TCP (6)	80	0.0.0.0/0	Allow
110	HTTPS (443)	TCP (6)	443	0.0.0.0/0	Allow
120	Custom TCP	TCP (6)	1024-65535	0.0.0.0/0	Allow
*	All IPV4	All	All	0.0.0.0/0	Deny

Table 3.3.1. Public subnet NACL inbound traffic rules

- **Rule 100:** Allows inbound HTTP traffic on port 80 towards any IPv4 address on the internet.
- **Rule 110:** Allows inbound HTTPS traffic on port 443 towards any IPv4 address on the internet.
- **Rule 120:** Allows returning TCP traffic from the internet responding to requests from the subnet. The specified port ranges are ephemeral ports as defined by the Internet Assigned Number Authority or IANA and Internet Engineering Task Force or IETF in their Request for Comments or RFC 6056 document [9].
- **Rule *:** Block every other non previously evaluated IPv4 traffic.

Outbound traffic					
Rule	Type	Protocol	Port range	Destination	Allow/Deny
100	HTTP (80)	TCP (6)	80	0.0.0.0/0	Allow
110	HTTPS (443)	TCP (6)	443	0.0.0.0/0	Allow
120	Custom TCP	TCP (6)	1024-65535	0.0.0.0/0	Allow
*	All IPV4	All	All	0.0.0.0/0	Deny

Table 3.3.2. Public subnet NACL outbound traffic rules

The rules explanation are similar to those for inbound traffic in table 3.3.1, except that instead of inbound it is outbound.

3.3.2 Private subnet

Most of the infrastructure components are deployed in the private subnet where only specific traffic from the public and isolated-private subnets are allowed in. In this subnet is where the UAV command and control center user interface is deployed, in containers using the AWS Fargate serverless service. The rules for this subnet have to be carefully defined so that;

- Fargate services can pull docker images from docker hub public repositories on the internet.

- The UAV, and several command and control application services can talk to each other.

Table ... and ... show the inbound and outbound traffic rules respectively configured on the private subnet network access control list or NACL.

Inbound traffic					
Rule	Type	Protocol	Port range	Source	Allow/Deny
100	HTTP (80)	TCP (6)	80	0.0.0.0/0	Allow
110	HTTPS (443)	TCP (6)	443	0.0.0.0/0	Allow
120	Custom TCP	TCP (6)	1024-65535	0.0.0.0/0	Allow
130	Custom TCP	TCP (6)	3306	10.0.4.0/28	Allow
140	Custom TCP	TCP (6)	3306	10.0.5.0/28	Allow
*	All IPV4	All	All	0.0.0.0/0	Deny

Table 3.3.3. Private subnet NACL inbound traffic rules

3.3.3 Isolated-private subnet

Describe the solution on a higher level. Discuss HLDs.

Chapter 4

Setup

4.1 Software

4.1.1 Command and Control Center

4.1.2 Telemetry dashboard

4.2 Hardware

Chapter 5

Discussion

Chapter 6

Conclusion

6.1 Future Work

6.1.1 Low latency communication

6.1.2 Collision avoidance navigation

Bibliography

[1] V. PRISACARIU, "The history and the evolution of uavs from the beginning till the 70s – doaj", *JOURNAL OF DEFENSE RESOURCES MANAGEMENT*, vol. 8, pp. 181–189, 1 Apr. 2017, ISSN: 2068-9403. [Online]. Available: https://doaj.org/article/7644116bc43d48a9ab450a50196b179a.

[2] J.-L. Samaan, "The proliferation of drones in the middle east has done nothing to change the strategic order there", *Orient XXI*, 2022. [Online]. Available: https://orientxxi.info/magazine/the-proliferation-of-drones-in-the-middle-east-has-done-nothing-to-change-the,5393 (visited on 05/15/2022).

[3] M. G. Levy, "Drones have transformed blood delivery in rwanda", *WIRED*, 2022. [Online]. Available: https://www.wired.com/story/drones-have-transformed-blood-delivery-in-rwanda/ (visited on 05/15/2022).

[4] W. Africa, "Covid-19 response in rwanda: Use of drones in community awareness | who | regional office for africa", *World Health Organization*, Jul. 2020. [Online]. Available: https://www.afro.who.int/news/covid-19-response-rwanda-use-drones-community-awareness.

[5] D. Pedro, J. P. Matos-Carvalho, F. Azevedo, *et al.*, "Ffau—framework for fully autonomous uavs", *Remote Sensing 2020, Vol. 12, Page 3533*, vol. 12, p. 3533, 21 Oct. 2020, ISSN: 2072-4292. DOI: 10.3390/RS12213533. [Online]. Available: https://www.mdpi.com/2072-4292/12/21/3533/htm%20https://www.mdpi.com/2072-4292/12/21/3533.

[6] A. W. Services, *What is aws*, 2022. [Online]. Available: https://aws.amazon.com/what-is-aws/.

[7] A. W. Services, *Aws pricing*, 2022. [Online]. Available: https://aws.amazon.com/pricing/.

[8] B. Gozluklu and R. Gupta, *Automating wind farm maintenance using drones and ai | aws for industries*, Jul. 2021. [Online]. Available: https://aws.amazon.com/blogs/industries/automating-wind-farm-maintenance-using-drones-and-ai/.

[9] M. Larsen and F. Gont, *Recommendations for Transport-Protocol Port Randomization*, RFC 6056, Jan. 2011. DOI: 10.17487/RFC6056. [Online]. Available: https://datatracker.ietf.org/doc/html/rfc6056#section-2.