An Investigation into the security of ACARS and other aeronautical datalinks

By

Matthew Boyce

BSc (Hons) Computer Science (Cybersecurity)

Staffordshire University

A project submitted in partial fulfilment of the award of the degree of BSc(Hons) Computer Science (Cybersecurity) from Staffordshire University.

Supervised by

Tomasz Bosakowski

Assessed by

Chris Mountford

May 2022

An Investigation into the security of ACARS and other Aeronautical Datalinks

Matthew Boyce

B018567J 09/11/2021 COMP60024

Table of Contents

1. Introduction

1.1 - Background and Context

History of Aeronautical Datalink Systems

During the 1970's, airlines and management bodies wanted a means to track pilot and aircrew working hours better than self-reports or written logs. These methods were unreliable and prone to manipulation - intentional or unintentional. The Aeronautical Radio Incorporated company, or ARINC, was initially approached to develop a system to automate the logging process, utilising a collection of sensors onboard the aircraft to determine the phase of flight. This gave rise to something known as an OOOI report (Out of gate, Off ground, On ground, Into gate). These four elements represent the four key stages of a flight. (ARINC, 2002).

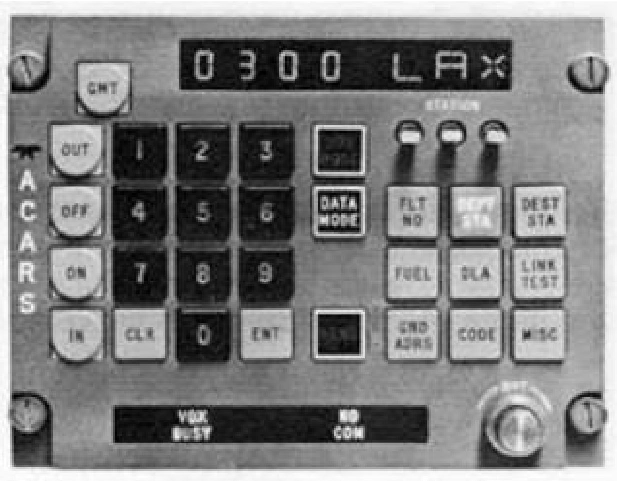
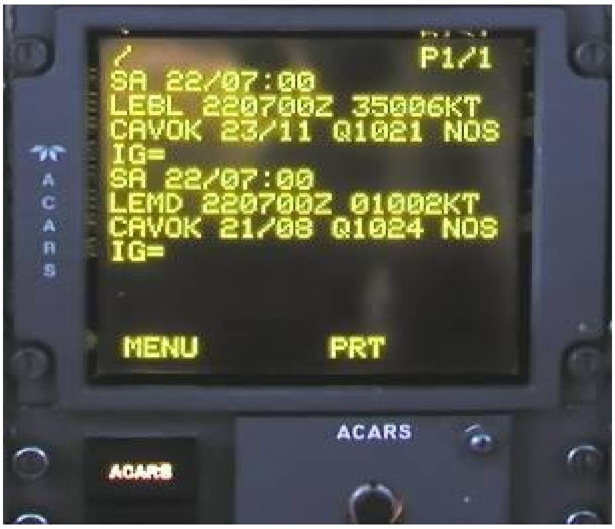


Figure 1 - Original ACARS Control Panel, Circa 1970 (ARINC, n.d.)



```
                         P1/1
SA 22/07:00
LEBL 220700Z 35006KT
CAVOK 23/11 Q1021 NOS
IG=
SA 22/07:00
LEMD 220700Z 01002KT
CAVOK 21/08 Q1024 NOS
IG=
```

Figure 2 - Modern ACARS Panel on a Boeing 737 (The Boeing 737 Technical Site, n.d.)

The system designed by ARINC would relay OOOI reports over a Very High Frequency (VHF) radio link to ground stations near major airports, using the existing aircraft radios used for voice communications.

The system rapidly took off and became commonplace within commercial aviation. As it grew, additional features were implemented, such as the ability to send text-based weather reports, and other information about the state of a flight.

These messages are known as Airline Operational Control (AOC) and Airline Administrative Control (AAC) messages and are focused primarily on the strategic management of flights from an airline and passenger perspective. Typical information includes the gate number of the arrival gate, the state of connecting flights or passenger-related information. AOC and AAC messages are not regulated, aside from the need to adhere to the ARINC 623 messaging standard, to maintain compatibility with the wider system. #REF.

In 2022, ACARS is a standard datalink protocol (ARINC 618) that facilitates datalink communication between an airborne entity and a ground network, utilising traditional radio and Satellite communications as a transmission medium. (ARINC, 2014)

1.2 - Scope

The scope of the project revolves around the provision and implementation of existing aeronautical datalinks, and their security attributes. These datalinks, as discussed within the report, are comprised of VHF data messages, operating within a publicly accessible frequency band. These signals form the basis of the ACARS messaging system, as well as the more modern CPDLC system. Signals that operate on a satellite or HF basis are beyond the scope of this report, due to the hardware complexity and cost required to interpret them.

Additional elements of the datalink system, such as airborne processing networks or ground receiver networks, are beyond the scope of this report due to their proprietary nature, and inability to test or develop solutions applicable to these systems. In addition, the threat models for these systems are considerably different, as these are inaccessible to the public.

1.3 - Project Plan and Management

In order to effectively manage the project, a suitable project planning and management strategy is needed. The scope of this is discussed further in this section.

Based off initial research, the field of aeronautical datalinks is undergoing little major change. As a result of this the requirements for any solution or implementation in this field are likely to remain static or slow changing. Considering the complexity of the field, a waterfall approach to the overall management of the project is likely to yield the best results. The project will be split into distinct categories, with each category forming the stages of a waterfall implementation.

This can be seen in the project plan, attached as an appendix to this report. In general, the project is split into multiple project units - each 2 weeks long. This was done in an attempt to make it easier to keep track of the project deliverables.

1.4 - Investigation Method

The investigation stage of the project is one of the most significant and in-depth elements. As a result of this, and considering the complexity of the problem domain and the problem identified, a suitable strategy needs to be defined in order to ensure all aspects are duly considered.

The nature of the problem within the report, focusing primarily on processes and systems, requires an investigation method with a qualitative focus, rather than a quantitative one. According to an analysis of research methods (Marzanah A.Jabar, 2009), qualitative investigative methods utilise observation and documentation to gather information and determine conclusions, rather than quantitative data. Whilst there is an element of quantitative data associated with the problem, the primary focus is qualitative.

An analysis of investigation methods within computing, (Nana Yaw Asabere, 2014), outlines methodologies that can be employed to problems within the field of computer science. Of these methods, a number apply to the problem domain of the project:

- *"Build Methodology"*
  The Build methodology revolves around building an artefact or solution to demonstrate the possibility of such an artefact. For the artefact to be effective, it must include novel features not found in similar implementations. (Nana Yaw Asabere, 2014)

- *"Experimental Methodology"*
  The Experimental Methodology can be divided into two phases - exploratory and evaluation. The first phase revolves around exploring an idea, concept, or solution, and gathering information that will enable them to determine further analytical directions relating to the concept. The second stage, evaluation, will revolve around the efforts to answer the questions that were determined in the previous stage. (Nana Yaw Asabere, 2014).

Considering the analysis performed above, the following research strategy can be defined:

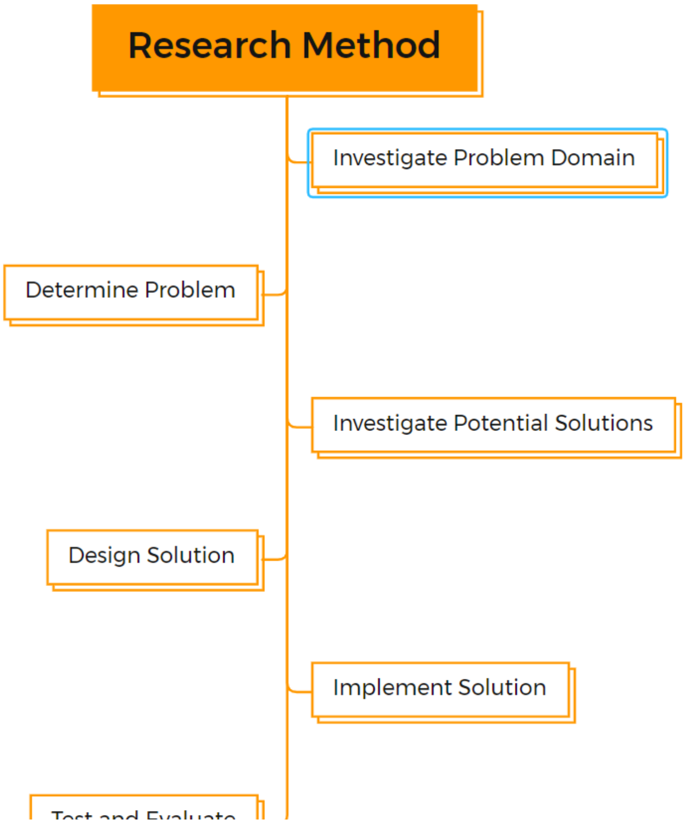


Figure 3 - Overview of Research Method

Figure 3 shows the strategy used for the project. The first element will revolve around investigating the problem domain. This will involve background research, a review of the existing literature relating to the problem domain and determining the scope.

The next section will utilise the information gathered in the first section to determine the actual problem to solve.

The next stage will revolve around analysing the problem defined in the previous section and determining a list of potential solutions to that problem. These potential solutions may relate to previously attempted external works, and the viability of these will be assessed before determining the end solution.

The next stage relates to implementing the solution determined to be most effective in the previous stage. This relates to the "build" section of the investigation method, defined above.

The final section will revolve around determining the effectiveness of the implemented solution, and therefore the overall degree of viability.

1.5 - Objectives and Success Criteria

The overall objectives of the project are to perform an investigation and analysis of the problem domain, and through this investigation create an exploratory solution that not only addresses the issues determined in the design and analysis stages, but also determines the viability of the solution in general, in addressing the problem identified. Once this is complete, the solution should be evaluated to establish limitations and directions for future work, should the project be extended.

#	Stage	Objective	Description
1	Investigation	Perform Background Research	Perform background research into the domain, get an understanding of major themes, works and attributes.
1.1		Analyse Existing Works	Analyse existing works to identify gaps in research and research directions across the field
1.2		Analyse Existing Solutions	Use existing works to identify existing solutions to the problem that may be in place, or where issues lie.
2	Determine Problem/Analysis	Determine problem	Through research, define and determine the exact problem to be solved.
2.1		Define Problem Scope	Determine which elements of the problem should be addressed, and which elements are beyond the reasonable reach of the project, or may fall under different areas.
2.2		Define Problem Components	Using the scope, split the problem into component parts. These parts will help to form the solution
2.3		Impact Analysis	Determine the impact of specific cybersecurity risks discovered in the investigation and analysis
2.4		Threat/Risk analysis	Perform threat and risk analysis of the problem domain.

3	Investigate Solutions/Design	Create list of potential solutions	Using a design or investigation process, determine some prototype solutions to the problem that may solve the issues addressed. These solutions may not be suitable, but the generation process will
		Determine viability and impact of potential solutions	Through analysis, determine the impacts, viability and overall viability of each potential solution
		Select solution with greatest viability	After analysis, determine which solution is the most viable, and therefore the solution to implement
4	Implement Solution	Use design to create a list of objectives	Using the design, create a list of objectives that the implementation should meet in order to adhere to the design and effectively address the problem
		Use objective list to create solution	Using the list of objectives, create the implementation.
5	Test and Evaluate Solution	Evaluate the solution effectiveness	Through testing, determine the effectiveness of the solution at addressing the problem domain.
		Evaluate the solution against objectives	Using the objective list, determine how the solution performed at addressing these objectives.

2. Research and Investigation

2.1 - Introduction to Aeronautical Datalinks

As mentioned in the introduction, aeronautical datalinks are a collection of digital, wireless systems that are designed to allow aircrew and ground-based entities to exchange messages, based on the strategic or the operational management of an aircraft. The next sections of the report will focus on these systems, and the infrastructure that supports them.

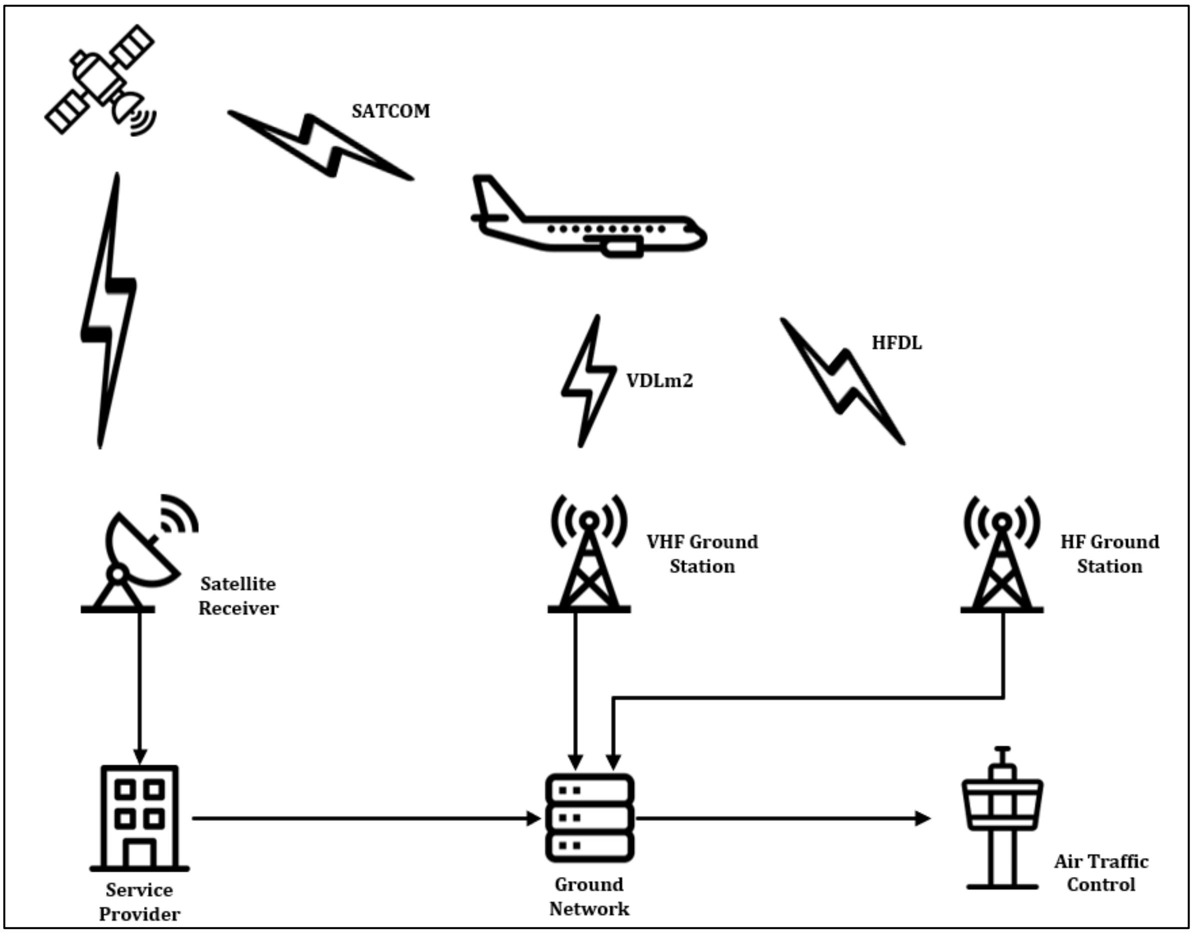


Figure 4 - Overview of Datalink Systems

2.2 - Physical Transmission Mediums

The most basic element of a datalink system is the method in which messages are passed from one entity to another. Due to geographical restrictions, flight requirements, and management body differences, there are several methods that are employed to this end.

2.2.1 - "Plain Old ACARS" (POA)

POA messages are the original implementation of ACARS messages. These utilise standard VHF voice radios and are transmitted using minimum-shift keying (MSK) modulation. POA messages have a throughput of approximately 2.4kbps and are employed in some areas for legacy or redundancy reasons, however they are increasingly being phased out in favour of more modern standards. (Haindl, 2007)

2.2.2 - VHF Datalink Mode 2 (VDLm2)

VHF Data Link mode 2, or VDLm2, is a newer format and has greater transfer speeds compared to POA. VDLm2 messages are one of the physical message formats used to support the Aeronautical Telecommunications Network (ATN), however lower levels of the VDLm2 protocol stack are employed separately to support ACARS communications. This implementation is known as "ACARS over AVLC[1]". (WAVECOM, 2022)

VDLm2 messages, including the ACARS provision is used primarily to support the ATN, as earlier message formats did not meet performance requirements. (ICAO, 2016). VDLm2 messages utilise differentially encoded 8-phase shift keying (D8PSK) as an encoding method and have a higher bit rate of 31.5kbps. VDLm2 uses a carrier-sense multiple avoidance (CSMA) protocol, also found within Wi-Fi, to prevent message collision. (ICAO, 2015)

[1] AVLC standing for Aviation VHF Link Control

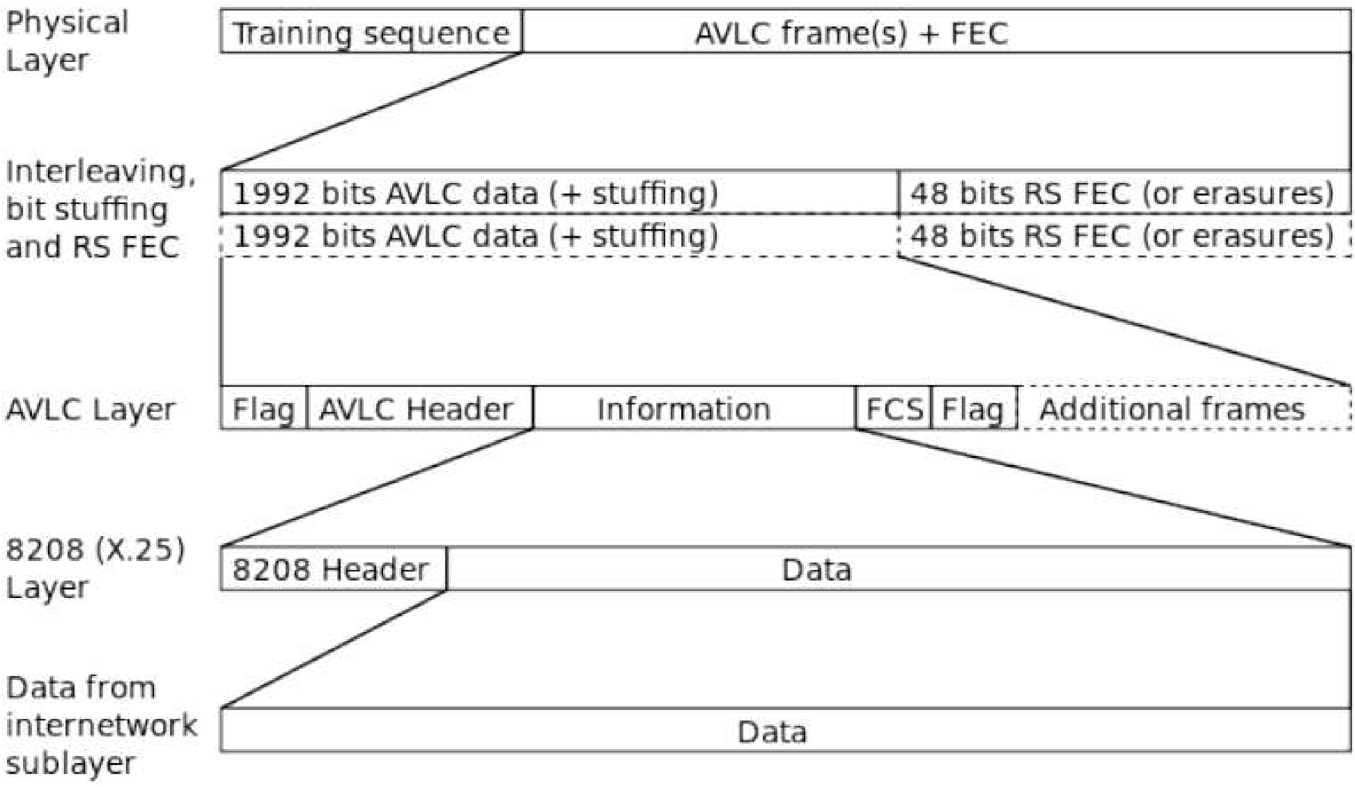

Physical Layer
Training sequence
AVLC frame(s) + FEC
Interleaving, bit stuffing and RS FEC
1992 bits AVLC data (+ stuffing)
48 bits RS FEC (or erasures)
1992 bits AVLC data (+ stuffing)
48 bits RS FEC (or erasures)
AVLC Layer
Flag
AVLC Header
Information
FCS
Flag
Additional frames
8208 (X.25) Layer
8208 Header
Data
Data from internetwork sublayer
Data

Figure 5 - VDLm2 Message Structure (Lundstrom, 2016)

Figure 5 shows a breakdown of a typical VDLm2 message, with the raw X.25 frame encapsulated within the Aviation VHF Link Control (AVLC) frame before transmission. The ISO 8208 (X.25) protocol is used as part of the network layer within the VDLm2 protocol stack. (WAVECOM, 2022). The AVLC frame is appended with forward error-correction (FEC) data to aid in synchronisation between the transmitter and the receiver. An additional training sequence is also appended at the physical layer, consisting of a transmitter ramp-up section and a known training sequence, to further facilitate synchronisation between the transmitter and the receiver. (Lundstrom, 2016).

2.2.3 - Satellite Communications (SATCOM)

Satellite Communications, or SATCOM, are often employed in the absence of other communication methods. This is required in remote regions, such as flights traversing oceanic or high polar routes. Satellite communication is a newer transmission medium that is often used in conjunction with High-Frequency Datalink (HFDL) to ensure coverage in remote regions. (ICAO Inter-Regional SATCOM Voice Task Force, 2012)

Messages sent over SATCOM utilise either the Inmarsat or Iridium constellations. Messages are transmitted from the aircraft to a satellite, where they are then passed to a ground receiving station, before being forwarded on to traditional datalink networks. For uplink messages (between an aircraft and a satellite), transmitters use the “L-Band” frequency of 1.5Ghz, within the UHF ITU band. This is due to the smaller size requirements for transmitters, as these are mounted within the aircraft.

For downlink messages (between a satellite and ground infrastructure), the 3.5Ghz C-band is used, as this permits greater bandwidth at the cost of signal path loss. Ground stations do not have the same size constraints as airborne hardware, and larger antennae can be used to compensate for signal path loss.

2.2.4 - High-Frequency Datalink (HFDL)

High-frequency data link (HFDL) messages are transmitted over the high-frequency (HF) ITU band. The HF band is defined by the ITU as between 3 and 30Mhz. (ITU, 2000) Unlike VHF, which is limited to line-of-site (LOS) range, HF signals utilise the ionosphere to "bounce" back down to earth. This allows them to travel beyond line of site (BLOS), and often beyond the horizon. This phenomenon is known as "skywave propagation", and enables signals to travel more than 5000km, depending on atmospheric conditions. (James V. Harmon, 1984)

HFDL messages are used in place of SATCOM messages in remote areas, where traditional VHF data links are out of range. In high polar regions, SATCOM communications can be difficult, due to the inclination of the satellites at high latitudes. In these cases, HFDL exists as a reliable means of maintaining datalink communications with an aircraft and a ground network.

2.3 - Transmission Hardware

Aeronautical datalinks, such as CPDLC and ACARS require a collection of aviation electronics (avionics), to operate. These avionics facilitate the transmission, reception, and processing of radio-based messages. The hardware type is closely linked to the transmission medium, with satellite communications requiring different avionics to HF communications.

The datalink network can be split into two main components - Airborne hardware, including space-based equipment such as satellites, and ground-based equipment, such as VHF, UHF and HF receiving stations, and the ground infrastructure that supports these stations.

The ground network used to distribute messages between stations and air traffic control is beyond the scope of this project.

2.3.1 - Airborne Hardware

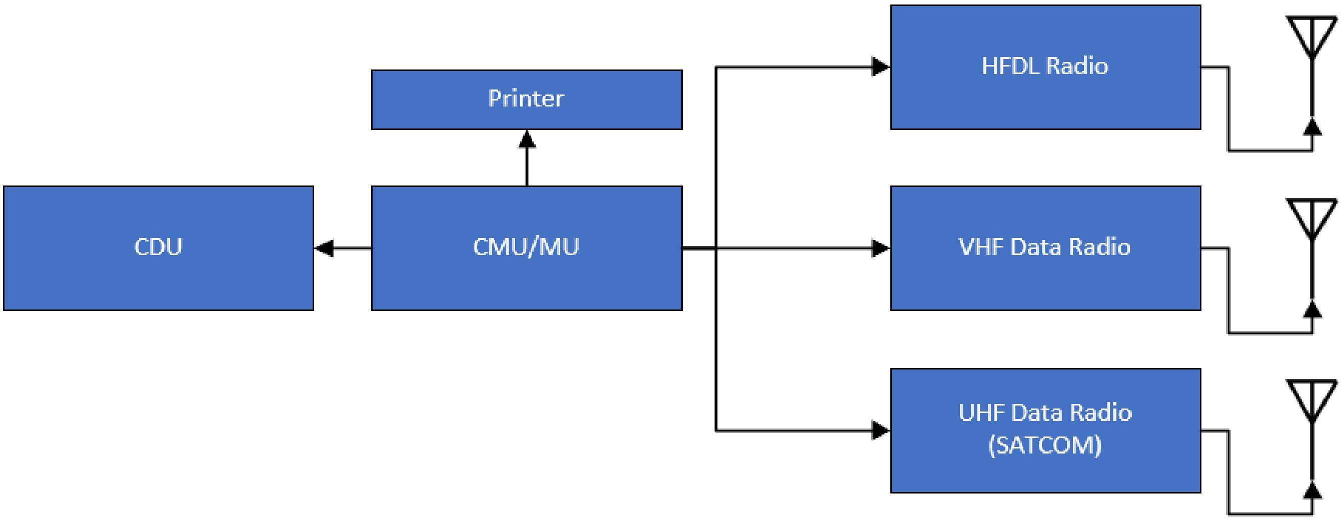


Figure 6 - Simplified diagram of airborne communications layout for datalink operation (Skybrary, n.d.)

Figure 6 shows a simplified version of a traditional datalink system inside a modern aircraft. Due to the complexity of these networks, and the potential for interfacing with a multitude of aircraft-specific subsystems and avionics, an exact representation of these systems unrepresentable.

- **CDU - Cockpit Data Unit**

  The first block, the CDU, refers to the entity within the cockpit for the aircrew to view or create messages. This is occasionally combined with other avionic devices, such as a flight management computer (FMC).

- **CMU/MU - Communications Management Unit/Management Unit**

  The CMU is responsible for routing messages through the connected systems. In older aircraft a simple “management unit” is used for solely ACARS connectivity, however in more modern aircraft a combined system that handles HFDL, SATCOM, and VDLm2 messages is used, known as a Communications Management Unit, or CMU.

  In more modern aircraft, the CMU will determine the best transmission format based on several attributes, such as signal strength, location and message content.

- **Data Radios**

  The data radios are designed to transmit and receive messages for their specific subnetwork.

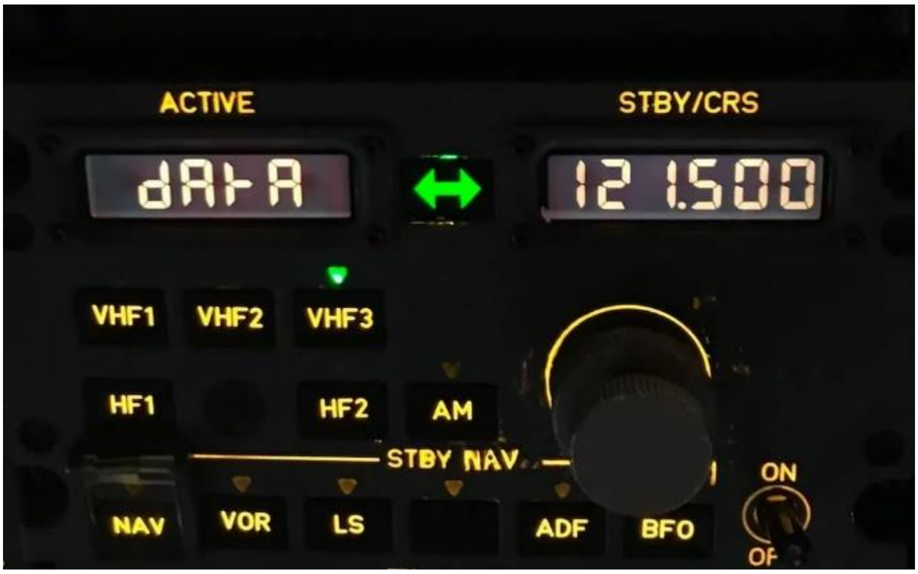


Figure 7 - Radio control panel in a modern aircraft, showing the transmission of a data message

In some cases these radios will be combined with the radios used for traditional voice communication, or a completely separate entity. Figure 7 shows a radio control panel in an Airbus aircraft, with data transmission in progress. Below the screen are the selector buttons for the VHF transmitters. (Airbus, 1999)

- **ARINC 429/629/818 Data Busses**

  The ARINC 429, 629, and 818 standards define the data busses in use in modern aircraft. These data busses are used to transmit data to and from various devices within the aircraft. (A. Kaviyarasu, n.d.). ARINC 429 is the oldest standard, with ARINC 629 and 818 bringing a number of improvements, such as transmission speed.

  These data busses are key components in a datalink system. ACARS and CPDLC messages will likely utilise these data busses when transferring from the cockpit interface to the management unit or data radio.

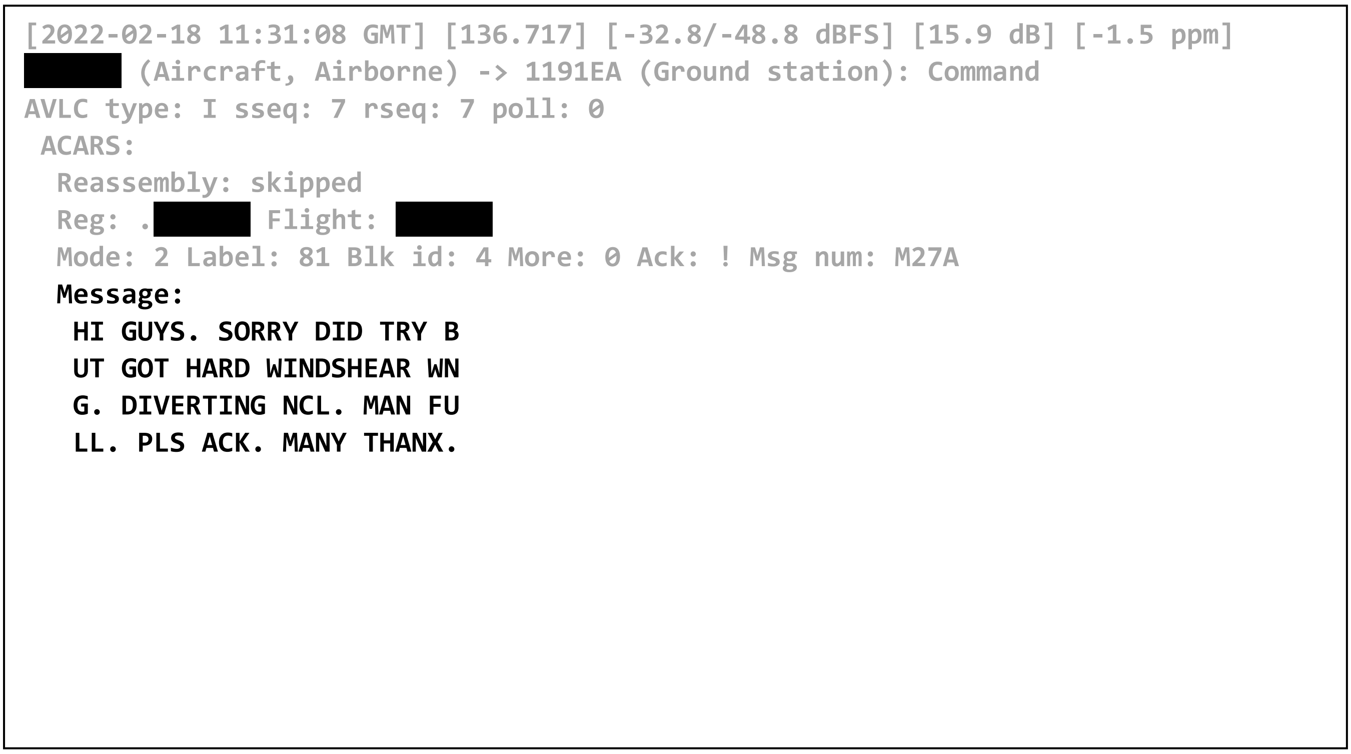
2.4 - Aircraft Communication and Reporting System (ACARS)

ACARS, as a system, is an implementation of the Aircraft Communication and Reporting System, designed in the 1970's for strategic and operational management of aircraft. ACARS shares a name with the original implementation, also known as VHF Data Link Mode 0 (VDLm0), as initially there was only a single datalink protocol associated with the system.

An analogy of this would be the difference between the low-level protocol used to send SMS messages, and the messaging application itself.

From a system point of view, ACARS as a system operates in a similar manner to text messaging, with aircrew and ground controllers able to exchange messages. These messages relate to the strategic management of aircraft from a controller's point of view (ATS), such as heading, altitude, or frequency change instructions, as well as from an airline operational point of view (AOC), with information about receiving gate and connecting flights.

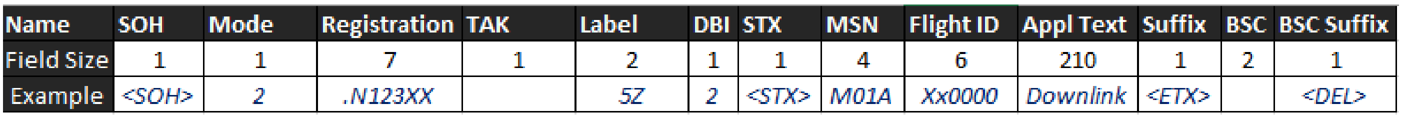
In a modern setting, ACARS now refers to a particular system used to transmit messages between an aircraft entity and a ground entity, over a multitude of transmission mediums.



```
[2022-02-18 11:31:08 GMT] [136.717] [-32.8/-48.8 dBFS] [15.9 dB] [-1.5 ppm]
[redacted] (Aircraft, Airborne) -> 1191EA (Ground station): Command
AVLC type: I sseq: 7 rseq: 7 poll: 0
 ACARS:
  Reassembly: skipped
  Reg: .[redacted] Flight: [redacted]
  Mode: 2 Label: 81 Blk id: 4 More: 0 Ack: ! Msg num: M27A
  Message:
   HI GUYS. SORRY DID TRY B
   UT GOT HARD WINDSHEAR WN
   G. DIVERTING NCL. MAN FU
   LL. PLS ACK. MANY THANX.
```

Figure 8 - Example of an ACARS message captured using DumpVDL2

2.4.1.1 - ACARS Message Format and Composition



Name	SOH	Mode	Registration	TAK	Label	DBI	STX	MSN	Flight ID	Appl Text	Suffix	BSC	BSC Suffix
Field Size	1	1	7	1	2	1	1	4	6	210	1	2	1
Example	*<SOH>*	*2*	*.N123XX*		*5Z*	*2*	*<STX>*	*M01A*	*Xx0000*	*Downlink*	*<ETX>*		*<DEL>*

Figure 9 - ACARS Message Format (Sun, 2009)

Message Element	Description
SOH (Start of Header)	This field identifies the start of the message header.
Mode	Mode category of the message. ACARS messages can be one of two main modes - Mode A and B. Mode A

	messages are signified by a "2" in this field, with category B messages signified with a symbol - between "@" and "]". The specific symbol is dictated by the ground network. (Sun, 2009)
Registration	The aircraft registration number, e.g "G-MARA" or "N123XX".
TAK (Technical Acknowledgement)	A technical acknowledgement requirement
Label	A two-character field that is used to describe the contents of the message
STX (End of Preamble)	-
Text	220-character text field (ISO-5 Encoded)
MSN (Message Sequence Number)	Used by ground processing network to reassemble fragmented messages
Flight ID	Two character airline identifier and four character flight number field (Sun, 2009)
Suffix	Used in multi-block messages to dictate end of message sequence or end of message component
BSC (Block Check Sequence)	16-bit field used in error checking
BCS Suffix (Block Check Sequence Suffix)	Due to the modulation used within ACARS, this needs to be populated in order to facilitate decoding of the final element of the message (Sun, 2009)

Table 1 - ACARS Message Fields and Descriptions (Sun, 2009)

2.5 - Controller-Pilot Datalink Communications (CPDLC)

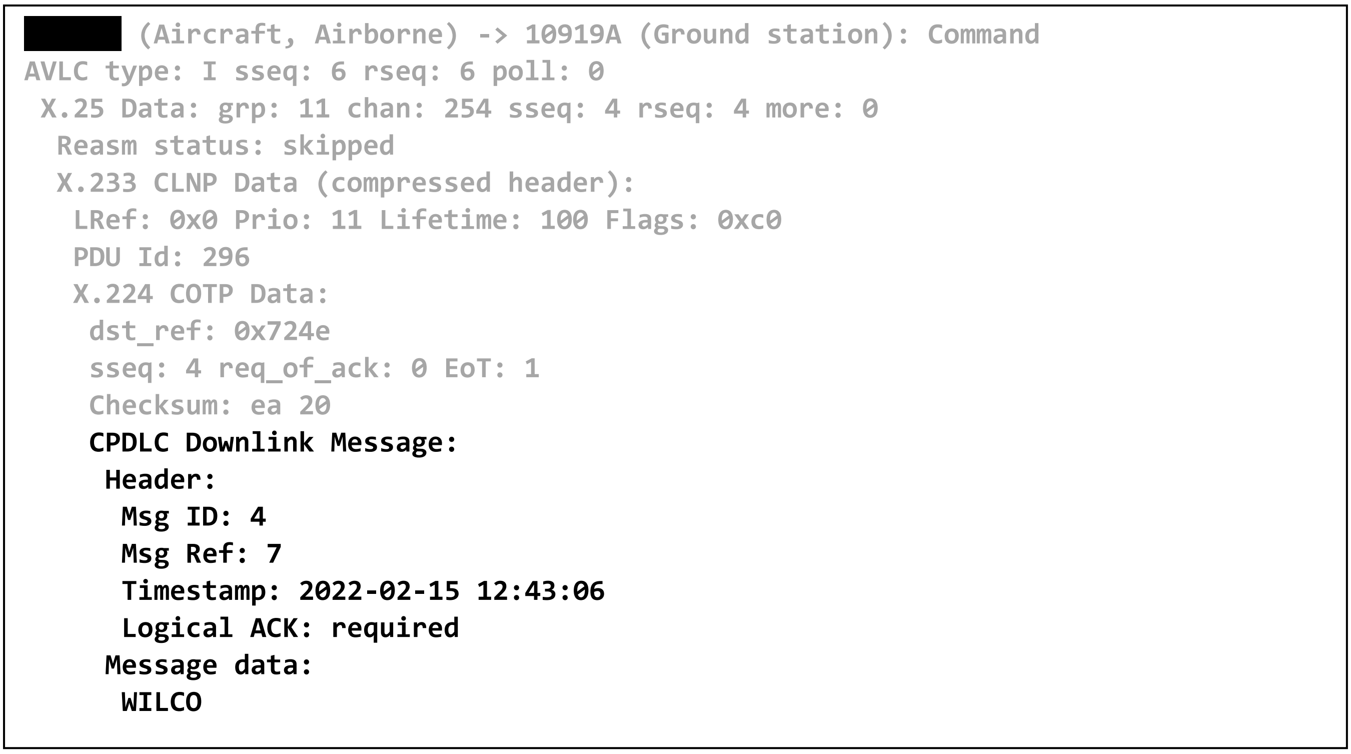
Controller-pilot datalink communications, or CPDLC, is "a datalink application that supports the exchange of data messages between a controller and flight crew" (ICAO, 2016).

CPDLC is designed to reduce the workload of air traffic controllers in areas of high traffic density, through the use of text-based messages instead of traditional voice instructions. This takes a fraction of the amount of time to complete, and also has the added benefit of reducing the chances of mis-interpretation due to poor signal quality or language barriers.

CPDLC is primarily used for the transmission of clearance and other traffic management applications, such as altitude adjustments or other aircraft commands. In addition to the standard message set, aircrew can transmit "free text" messages that do not follow a specific format.

CPDLC messages are made up of elements. Messages can be single element, in the sense that they contain a single instruction, or can be made up of multiple elements, containing multiple instructions. In addition to this, CPDLC messages can be transmitted with a “response attribute” - requiring the flight crew to respond to the message, either confirming the action (Roger/Wilco/Affirmative) or rejecting the action (Negative/Unable). (ICAO, 2013)

CPDLC messaging utilises similar datalink protocols used within ACARS. CPDLC utilises VDLm2, SATCOM, or HFDL, depending on flight characteristics.



```
[redacted] (Aircraft, Airborne) -> 10919A (Ground station): Command
AVLC type: I sseq: 6 rseq: 6 poll: 0
 X.25 Data: grp: 11 chan: 254 sseq: 4 rseq: 4 more: 0
  Reasm status: skipped
  X.233 CLNP Data (compressed header):
   LRef: 0x0 Prio: 11 Lifetime: 100 Flags: 0xc0
   PDU Id: 296
   X.224 COTP Data:
    dst_ref: 0x724e
    sseq: 4 req_of_ack: 0 EoT: 1
    Checksum: ea 20
    CPDLC Downlink Message:
     Header:
      Msg ID: 4
      Msg Ref: 7
      Timestamp: 2022-02-15 12:43:06
      Logical ACK: required
     Message data:
      WILCO
```

Figure 10 - Example of a CPDLC Message captured using DumpVDL2

2.6 - Literature Review

2.7 - Comparable Systems

The purpose of this section is to analyse similar systems and protocols in use in the industry, and similar areas. This will provide context and other use cases in which to draw comparison.

2.7.1 - Automatic Identification System (AIS)

The Automatic Information System (AIS) is a naval communication system that utilises digital radio signals for position reporting and collision avoidance amongst vessels. The system operates in a similar means to the ADS-B system utilised by aircraft for collision avoidance, with an entity repeatedly broadcasting location updates to other entities in the vicinity. AIS is required for vessels over a certain size, as well as all passenger carrying vessels, by the International Maritime Organisation (IMO). (International Maritime Organisation, 2018)

The system operates through automatic position reports by vessels equipped with AIS. The reports include the current location of the vessel, as well as speed and heading. This information is received by all vessels within range and is plotted on a map or chart. This allows vessel captains to maintain positional awareness and serves as a backup to marine radar.

In a similar way to systems used within aviation, AIS currently uses unencrypted and unauthenticated messages. Whilst message encryption in a system primarily designed for general reception by unknown parties may be detrimental to the performance of the system, the issues of message authentication and integrity are critical elements. In 2021, AIS messages presumably sent by British and Norwegian warships indicated that they were within Russian territorial waters, sparking conflict between the involved parties. Other sources, such as live-view cameras, indicated that the vessels in question were currently in port, and that the AIS messages had been spoofed. (Euronews, 2021)

Whilst this type of attack is easy to mitigate against, offsetting a ships position by a few miles, or injecting “ghost” ships into the network is likely to cause disruption and navigation issues for vessels in the area.

2.7.2 - Automatic Dependent Surveillance - Broadcast (ADS-B)

The Automatic Dependent Surveillance – Broadcast system, also known as ADS-B, is an aeronautical system designed to facilitate position reporting through the repeated broadcast of an aircraft’s position, heading and speed. ADS-B broadcast messages are designed to be received by ground radar installations, for the purposes of traffic management, and by other aircraft. Other aircraft in the vicinity of the transmitting aircraft will therefore be able to determine their proximity to the aircraft, and alert pilots if there is risk of a collision.

ADS-B is also faced with a similar set of problems as AIS. Because ADS-B signals are unauthenticated and unencrypted, there is the potential for a malicious entity to inject "ghost" aircraft into airspace. (Prince, 2012). If this occurs in a heavily congested area, it could cause an aircraft's Traffic Collision Avoidance System (TCAS) to engage. If engaged, an immediate instruction is issued to the pilot of an aircraft, advising the heading to turn to, and the altitude to climb or descend to, to avoid an imminent collision. In congested airspace where separation between legitimate aircraft is reduced, this could have serious consequences.

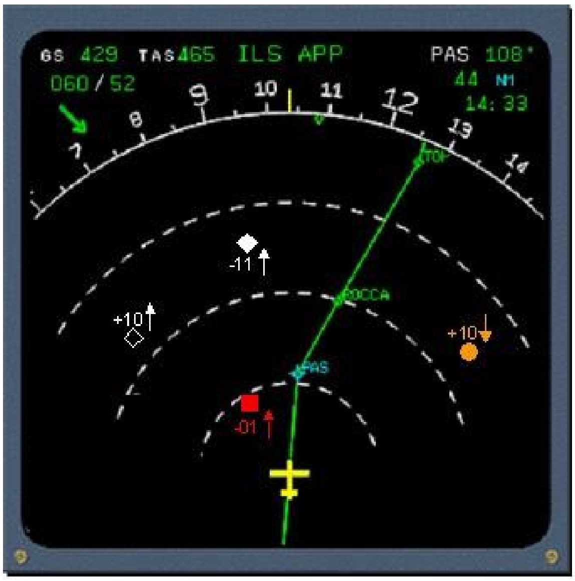


Figure 11 - TCAS system displayed on an EHSI Display

Figure 11 shows the TCAS system implemented on an Electronic Horizontal Situation Indicator (EHSI) display in the cockpit of a modern aircraft. The coloured symbols indicate potential collision risks.

2.7.3 - Satellite-based Communications

Satellite communication in general is a widely used element in modern digital communications. Satellite communications often face the same issues faced within the field of aeronautical datalinks. Namely, limited bandwidth, tight hardware limitations and high cost of upgrade/modification to existing hardware. Additionally, the security of existing and legacy satellite communications is threatened by the advent of software defined radio (SDR) devices, in a similar manner to aviation. These devices allow wide-band reception and transmission for a fraction of the cost of a dedicated hardware transceiver.

In addition to this, legacy satellites and satellite networks use customised or proprietary encryption protocols, which may not have endured the test of time. These methods may be vulnerable to modern processing power and other technological enhancements within the field of encryption.

2.7.3.1 - SATCOM Security Challenges

There are several non-trivial challenges faced by the industry in relation to security. High upgrade cost and technical limitations within legacy systems, often installed within the satellite itself, make it difficult to implement security patches.

In a similar manner to aeronautical datalinks, there also lies the issue of transmitting secure material (such as encryption keys and security updates) across an inherently secure medium, such as a wireless data link. Whilst many methods exist to this end (Diffie-Helman), many of these will not work on older hardware and across limited bandwidth.

Another challenge faced by this industry, is the fact that it is difficult to create a unified threat model, as the purpose, scope and implementation of each mission varies wildly. Because of this, dedicated security policies, implementations and frameworks may be required across each or small groups of missions. This makes ensuring and managing security in a changing landscape a daunting and expensive task. Furthermore, key management is also a critical issue within the field. In a large satellite constellation, there may be hundreds of satellites, across vast distances, and with poor-quality communication links between them. Coupled with the constant sign-on and sign-off as satellites pass in and out of range, key management becomes a significant challenge. This is also a significant challenge within aeronautical datalinks, with aircraft transitioning into and out of coverage zones.

2.7.4 - Comparable Systems Overview

In general, this section has identified the need for additional security measures in a number of similar industries that share common functionality with aeronautical datalinks. Potential security implementations in these systems are hampered by the requirement for openness (ADS-B, AIS), or the use of old or legacy hardware (Satellite communications).

2.8 - Security Analysis

The security of datalink systems is the primary focus of this project. The overall goal is to perform a security overview of these systems within the scope of the report, and then to utilise this security overview to attempt to solve the problem of security within aeronautical datalinks.

2.8.1 - Overview of Security within the field

As the aviation industry grows, the level of utilisation of existing communication networks and datalinks is likely to increase. This, in turn, will also increase the potential impact of any attack to these links. In addition to this, continued development within aviation, with a focus on increased automation, will likely increase the amount of trust given to datalink systems to automatically carry out certain tasks throughout the flight. As a result of these factors, the security of datalink systems such as CPDLC and ACARS is critically important.

The existing transmission systems utilised for datalink applications were designed in a time where outdated threat models were in place. (Martin Strohmeier, 2016). This threat model makes several assumptions that may not be accurate today, such as the implausibility of anyone outside a nation-state attacker being able to create, obtain, or access hardware suitable for interfacing with these datalinks.

This is no longer the case, with the advent of low-cost, readily available software defined radios (SDRs). As a result, datalink traffic can be intercepted by any party with a personal computer and suitable radio, permitting the interception, analysis, and decoding of messages sent across these bands.

It has been proven that existing systems used for datalink communication are vulnerable to spoofing or message injection attacks, using open-source SDR software. (Corentin Bresteau, 2018). Message spoofing is a particularly critical issue for CPDLC communications, as these messages are often interpreted directly by the flight crew, with little to no authenticity or integrity checks. This may be partly down to the notion that these messages originate from a trusted entity, i.e., ATC, and therefore no question needs to be raised relating to their validity.

Whilst the potential for an attacker establishing and maintaining control of an aircraft through this means is low, it does have a significant potential for disruption, especially in heavily congested airspace. If an attacker were to launch multiple spoofing attacks across multiple transmission mediums simultaneously, they could significantly impact the ability of air traffic control (ATC) to manage traffic. In a busy area such as the airspace around an international airport, this could potentially have disastrous consequences.

2.8.2 - Existing Security Implementations

According to a document published by the Airlines Electronic Engineering Committee (AEEC), there exists several relevant projects and standards relating to cybersecurity within the field of aviation. (AEEC, 2018) Within this document are several projects and standards that apply to the field of aeronautical datalinks. These include:

ARINC Specification 823 - ACARS Message Security + Key Management
This document details a secure implementation of ACARS, utilising asymmetric key cryptography to facilitate end to end message encryption.

ARINC Specification 822A - On-Ground Aircraft Wireless Communication
The focus of this document is to outline and implement guidelines for the use of commercial datalinks whilst an aircraft is on the ground, such as when parked at a gate. These guidelines cover IEEE802 and cellular telephony. (AEEC, 2018).

2.8.2.1 - ARINC 823 - Secure ACARS

The ARINC 823 standard was designed as a joint operation between the US Air Force and Honeywell (Matthew Smith, 2017), as a means for USAF aircraft to utilise civil routes that required ACARS compatibility, but without yielding potentially sensitive information, such as flight plan, location, and other telemetry that may be utilised by an adversary. ARINC 823 details ACARS Message Security (AMS), which uses end-to-end encryption, implemented at the application level. Symmetric or Asymmetric encryption methods can be employed, depending on the use case. Despite the protections involved with this system, little to no use of this has been identified in civilian air traffic.

Secure ACARS and Protected ACARS

Secure ACARS is the term given to an implementation of the ARINC 823 standard, through Honeywell international, whereas Protected ACARS is a more general offering by ARINC and other industry partners. (Matthew Smith, 2017)

2.8.2.2 - Proprietary Security Methods

In addition to ARINC 823, there also exist additional implementations of ACARS and other datalink security methods, notably those aimed towards the owners and operators of private aircraft. Little documentation or detail is publicly available, however a report investigating these indicates that the ciphers involved are weak and easily breakable. (Matthew Smith, 2017)

2.9 - Existing Works

A considerable amount of existing work has been done in security within aeronautical datalinks.

A study conducted (Joshua Smailes, 2021), investigates the security issues within the CPDLC systems in use today. An analysis is performed on the existing protocol, as well as the susceptibility of this protocol to attack. The study explores the potential impacts of a man-in-the-middle attack, through a practical data collection and analysis exercise.

The example attack explored in the study is designed to intercept and hijack the CPDLC communication from the target aircraft, through the imitation of a legitimate air traffic services unit, or ATSU. By transmitting a request to switch to the next data authority, which requires no authentication, an attacker can convince the CPDLC system on the aircraft to interface with an imitation ATSU controlled by the attacker. This way, the attacker will be able to send messages from a position of implicit trust.

The study goes on to propose countermeasures to this attack, namely modifications to the way handovers are processed, and the potential inclusion of message signing or authentication.

Another study (Matthew Smith, 2018), investigates the security of the ACARS network itself. The report analyses the vulnerabilities within the network from a privacy-centric point of view and focuses on the use of ACARS by privacy sensitive parties, such as military, governmental, and business operators. The study indicates that significant breaches of privacy can be found through the analysis of ACARS messages, including the privacy of "blocked" aircraft - i.e., aircraft that have been removed from flight analysis websites and other public sources.
Mitigations proposed by this paper include the use of standardised encryption schemes, such as ARINC 823's ACARS Message Security (AMS).

An investigation performed (Corentin Bresteau, 2018) presents a threat analysis on the current use of datalink communication protocols in aviation, including the potential threat impact of an impersonation attack on both ACARS and the FANS1/A protocol. This line of inquiry concludes with interviews with airline pilots and other qualified individuals to establish the likely safety impact of some of the attacks listed in the paper, revealing one potential attack method that may significantly impact the safety of a flight.
The paper goes on to investigate the implementation impacts of ARINC 823's AMS, including the impact that full implementation would have on overall radio network saturation, a key consideration when determining a security implementation for these systems. In addition to this, optimisations are suggested for the existing AMS standard, to further reduce the impact implementation would have on existing datalinks.

A paper presented (Matthew Smith, 2017), investigates the proprietary encryption methods that are used to protect ACARS and other CPDLC messages in privacy-orientated groups, namely business aviation. The paper investigates the cipher used to secure these messages, finding that the existing proprietary implementation is based on a substitution cipher, and is easily broken. By cross-referencing messages decoded with public sources of "blocked" or "hidden" aircraft, an attacker can determine the party to which the messages belong, further impacting the overall privacy of the aircraft.

Another study (Martin Strohmeier, 2016), investigates the security of wireless air traffic control communications from a more general perspective, including attacks on other air traffic control systems, such as ADS-B, primary and secondary surveillance radar, multilateration and CPDLC. A threat model is created, and an analysis of existing countermeasures is performed, showing that many of the protections implemented already in these systems are not suitable or effective.

A study conducted in 2000, (Roy, 2000) explores the issue of ACARS security for military applications, namely for the use of ACARS by the United States Air Force (USAF). The study investigates the security issues relating to ACARS, and proposes a secure version of this system, known as Secure ACARS. This proposed version later goes on to become the basis of the ARINC 823 AMS standard.

2.9.1 - ICAO Cybersecurity Action Plan

According to a document created by the ICAO, a proposed action plan relating to cybersecurity has been created. (ICAO, 2020). This action plan is comprised of seven key pillars:

- International Cooperation
- Governance
- Effective Legislation and Regulations
- Cybersecurity Policy
- Information Sharing
- Incident Management and Emergency Planning
- Capacity Building, Training, and Cybersecurity Culture

The document details information regarding the plan to increase the level of cybersecurity awareness and number of controls within the industry, including the implementation of a “caISMS” - An information security management system designed for civil aviation, based on the ISO27000 family of standards for information security monitoring.

This action plan highlights the increasing awareness within the industry as a whole in regard to cybersecurity and the risks posed by attacks to the various systems within the industry.

2.9.2 - Security Challenges

In general, the aviation industry has a number of challenges associated with improving general security. For example, because of the increased focus on reliability rather than new technology or features, the hardware that supports digital infrastructure within aviation is considerably dated. This makes modern cryptographic, monitoring or other security methods difficult or impossible to implement.

In addition to this, the certification cost and process associated with avionic hardware is extensive. This is to ensure the highest standards of safety and reliability are achieved, as system failure in a critical element must be avoided at all costs.

Due to these factors, and coupled with the growth in the aviation sector as a whole, existing systems are being stretched beyond their design capacity, further increasing the potential for security issues.

2.10 - Encryption within Aeronautical Datalinks

2.10.1 - Overview of Encryption

Encryption is the term given to the process of converting information from a human-readable form, into an obfuscated, encoded format. Data that has been encrypted cannot be read or interpreted until it has been decrypted. (Kaspersky, n.d.). Encryption is one of the cornerstones of modern digital communications, and is used in everyday applications.

Encryption processes can be split into two main types: asymmetric, and symmetric.

2.10.1.1- Asymmetric Encryption

Asymmetric encryption revolves around the use of public and private keys. The sender of a message uses the public key of the receiver, which is freely available, to encrypt the message. Once this has occurred, only the receiver with the corresponding private key, can decrypt the message. (IETF, 2007)

Various methods exist for the transmission and sharing of public keys, such as Diffie-Helman.

2.10.1.2- Symmetric Encryption

Symmetric encryption is a simpler method of encryption, whereupon a secret key is used to encrypt information. This same secret key must then also be used to decrypt the message. Symmetric encryption relies on both parties having access to the same secret key. One of the primary challenges within the field of symmetric encryption is key management, as anyone with the key will have access to the data. (Aleisa, 2015)

2.10.1.3- Common Encryption Standards and Processes

There are a number of widely used encryption algorithms in use today. One of the most common is the Advanced Encryption Standard, or AES. AES uses the Rijndael algorithm, which is a block cipher capable of using 128-, 192-, and 256-bit keys. (Aleisa, 2015).

AES has been adopted as a standard encryption technique according to ISO/IEC 18033-3:2010. (ISO, 2010)

2.11 - Research Conclusion

In summary, aeronautical datalinks have been investigated. The origin and background context in which they operate, their intended purpose, and their security attributes have been investigated and analysed. Through this analysis, a potential gap has been identified, regarding the confidentiality, integrity, and availability of the information transmitted across these links. To this end, a literature review and analysis of existing work has been carried out, whereupon existing security work has been identified. Technical literature from within the field (ARINC standards) have also been investigated, with a view to determining the potential pitfalls and issues relating to the provision of aeronautical datalink security. In addition to this, unrelated systems that operate under similar conditions, or towards similar goals have been investigated, with a view to compare and contrast the potential solutions that exist in these systems, in order to determine the viability of implementing these security solutions within the field of aeronautical datalinks.

To conclude the research, a requirement for additional security on these links is present, with a solution required that not only protects the information transmitted across such a link, but also adheres to strict technical requirements, enforced by older or outdated hardware, software, and overall infrastructure.

3. - Analysis

3.1 - Problem Solving/Analysis Method

Considering the research completed above, and the complexity of the problem domain, a suitable problem solving, or analysis method is needed, to ensure each aspect of the problem is considered and no elements are missed when determining a solution to the problem. A methodological approach will also ensure that exploration of each area of the problem is fully carried out, without missing aspects that may prove crucial later.

3.1.1 - Six-Step Problem Solving Model

The six-step problem solving model is an organisational model proposed by Edgar Schein, and consists of six key stages:

Table 2 - Six Step Problem Solving Model (Free Management Books, n.d.)

#	Stage	Description
1	Problem Definition	Through analysis of the problem domain, determine the problem that requires a solution.
2	Determine Root Cause	Through analysis of the problem, determine the root cause.
3	Develop Solutions	Develop solutions to the problem, focusing on how each solution relates to the identified root cause
4	Select Solution	Evaluate the solutions created in the previous step and determine the best solution for the problem
5	Implement Solution	Utilise the proposed solution into an actual solution through practical application
6	Evaluate Outcome	Determine the success of the solution in solving the original problem

3.2 - Application of model to problem domain

As discussed in the introduction, regarding the 'build methodology' approach to solving a problem, the six-step problem solving model works well with the set requirements of the field and limited changing of deadlines and other project requirements. In addition to this, the six individual stages can be mapped to a waterfall model to further aid with the management of the solution.

3.2.1 - Problem Definition

Regarding the problem domain, the definition of the problem relates to the lack of protection or encryption within aeronautical datalinks. This is likely to leave any communications that occur over these channels open to interpretation from unauthorised parties. Therefore, the problem can be defined as:

A security flaw within the field of aeronautical datalinks, notably the system in which a ground entity can communicate with an airborne entity via text and data messages, allows an unauthenticated party to intercept and inject messages into the system, which may result in operational issues, with potential consequences that impact the safety of aircraft.

Symptoms of this problem include:

- Confidentiality breaches
  - Contents of messages can be interpreted by an unauthorised party
- Integrity Breaches
  - Unauthorised parties may be able to modify the contents of the messages between two entities
- Availability breaches
  - Unauthorised parties may be able to prevent the system from operating in the way it was intended, through denial-of-service (DoS) techniques.

3.2.2 - Root Cause Analysis

A root cause analysis, or RCA, is a systematic breakdown of the problem, with a view to determining the individual factors, processes or entities that contribute to the problem in question. The goal of RCA is to determine why an event occurred, in order to fix the processes that led up to the event, in order to prevent event reoccurrence. (James J. Rooney, 2004).

The root cause of the problem defined above can be technical:

- A lack of cybersecurity awareness within the industry
- Outdated threat models resulting in ineffective controls

Alternatively, the root cause could be consequential:

- An outdated system expanding beyond its means
- Developing technologies within the field of computing are adjusting the scope of what is possible and therefore creating new challenges or problems.

3.2.3 - Solution Development

In an abstract sense, several potential solutions exist for the specific problem. These may not be suitable for development into an actual solution; however they play an important part in the problem solving process, by ruling out solutions that are infeasible for one reason or another.

#	Solution	Effectiveness	Viability
0	Remove datalink capability from aircraft	VH	VL
1	Implement additional verification procedures	M	M
2	Implement encryption into the existing system	H	M
3	Ban RTL-SDR Devices	H	VL
4	Increased legal penalties	M	L
5	Increased cybersecurity awareness	L	H

Table 3 - Potential Solutions to the problem of aeronautical datalink security

3.2.4 - Solution Evaluation and Selection

Of the solutions proposed in the above table, a subset of these can be disregarded immediately due to infeasibility. For example, removing datalink capability from the aircraft would be highly effective in mitigating the potential for security risks as a result of these links, however the loss of this utility would result in issues relating to operational capability in areas of high air traffic, and the subsequent saturation of traditional voice radio links may introduce safety concerns within these areas. This makes that solution unviable.

In addition, banning the use of RTL-SDR devices or increasing the legal retributions relating to their use is likely to hinder development in this field, discouraging research and other investigation, whilst potentially having limited effect on malicious users, who are likely to continue using them. The lack of effectiveness and legal scope required make this infeasible.

Implementing protection measures within the existing system also contains limitations and challenges. Existing literature on this approach is plentiful, with challenges such as cryptographic overhead, key management, bandwidth available and infrastructure limitations hindering this approach. Despite these limitations, the challenges faced by this approach are within the realms of possibility.

Considering the options available, the most suitable solution would be to attempt to secure the existing system through the use of encryption. This has been explored in the existing literature and is likely the best way to secure these systems.

To reduce the potential impact to existing systems that rely on the datalink infrastructure to operate, rather than modifying the entire system to implement encryption, an encryption layer can be added instead. This has the main advantage of being able to retain the underlying format of the system, enabling compatibility to be maintained with existing infrastructure. This additional encryption layer should be implemented in as transparent a way as possible, to ensure compatibility is maintained with the wider infrastructure.

3.2.5 - Solution Implementation

To implement this solution, a number of factors must be considered. These relate to the infrastructure that the datalink systems operate within, as well as the dependant systems that rely on these datalinks.

3.2.5.1 - Infrastructure Considerations

One of the key infrastructure considerations that must be adhered to when designing a potentially new layer would be the upper and lower layers immediately adjacent. The physical layer that supports these datalink protocols, VDLm2, will need to have enough available bandwidth to accommodate the cryptographic overheads that are likely to result from the addition of an encryption layer. Additionally, the proposed encryption layer must be suitably resilient enough to be transmitted over a potentially weak/lossy datalink and must support operation across multiple endpoints. Key management will therefore be a critical element to the new system.

3.2.5.2 - Dependent Systems

The next element that must be considered is the layers that depend on the system to function. In this case, the application-layer entities and applications that utilise datalinks to operate. CPDLC and ACARS messaging are key elements of the system and will interface with the lower layers through the encryption layer in a transparent manner.

3.2.6 - Proposed Solution

In order to solve the problem of security within aeronautical datalinks, a solution revolving around the implementation of an encryption layer within the existing system is proposed. This encryption layer will encapsulate the ACARS and CPDLC data that traverses the system. This encrypted data will be carried by the existing VDLm2 message format to maintain compatibility with the wider system. This proposed solution will be discussed further in the next chapter.

4. Design

4.1 - Design Methodologies

Due to the complexity of the problem domain, a suitable design methodology needs to be selected. This design methodology will ensure that all aspects of the proposed solution are covered.

4.1.1 - Overview of design methodologies

There are several different methodologies that can be applied to the field of software development, and therefore are suitable for use within developing the solution to the problem of security within aeronautical datalinks.

#	Name	Overview
0	Agile Development	- Operates around iterative development cycle - Focus on delivery of small elements - Designed to cope with changing requirements throughout the product development journey. (Atlassian, n.d.)
1	Extreme Programming (XP)	- A subset of agile programming - Main focus relies on good communication between team members and client -
2	Waterfall Development (Adobe, n.d.)	- Utilises a linear sequencing approach, whereupon each stage follows the previous in a linear fashion - Each phase should be finished before the next phase begins - Works best when fixed dates, requirements, and outcomes are known - Can be inefficient, may suffer from *deadline creep* - Fluid requirements can hinder and make this method inefficient

4.1.1.1 - Agile Development

As referenced above, agile development is an iterative development methodology that operates around the evolutionary principle of iterative enhancement - that is, iterative improvements on a project or software base over a set period of time. (Williams, 2007)

It is suggested that agile development techniques perform best when a concrete plan is unsuitable, or there are many changes to the requirements throughout the design process. This is due to it's iterative nature, as modifications and changes to the requirements are not restricted by a set plan at the beginning.

4.1.1.2 - Waterfall Development

Waterfall development methodology is a traditional software design principle that revolves around sequential stages, each one coming after the other, originating from requirement definition and culminating in a final product.

Waterfall development is widely regarded as a rigid model that does not cope well with changing software requirements, a common occurrence within software development. Despite this, the waterfall method is still widely employed throughout software development. (Kai Petersen, 2009).

4.1.1.3 - Chosen Strategy

The chosen strategy, based on the attributes of the problem domain and problem definition, is the waterfall development strategy. The fixed requirements and outcomes of the problem definition make waterfall development ideally suited to a project of this nature.

4.2 - Solution Design

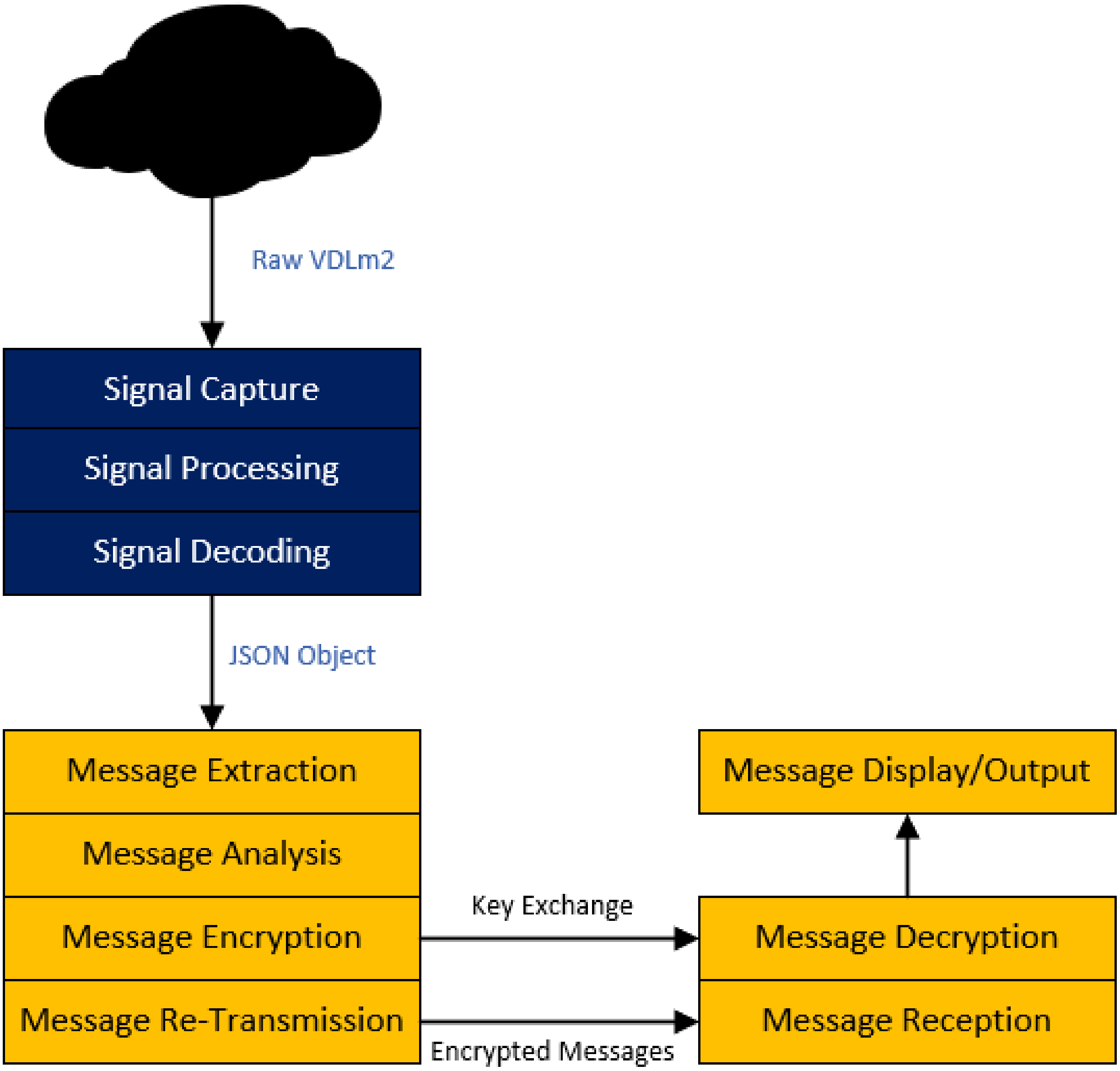


Figure 12 - Overview of Intended Solution

4.2.1 - Overview

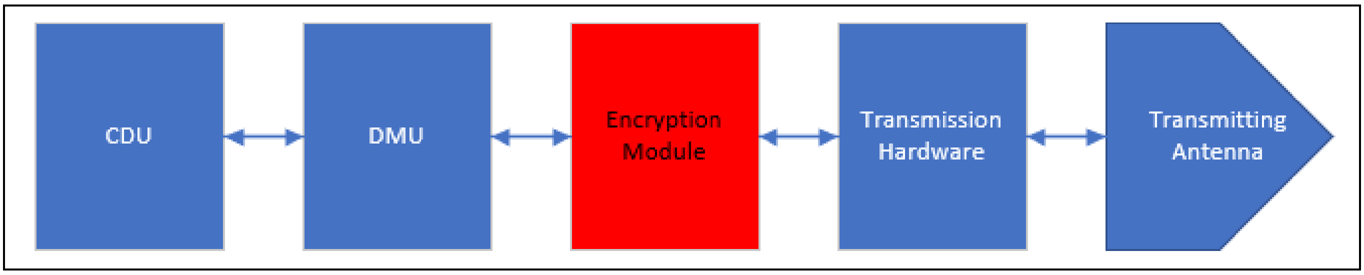


Figure 13 - Transmission Process Overview

Figure 13 identifies the general process a message goes through when being transmitted over either CPDLC or ACARS. This also highlights the location of the proposed encryption layer in relation to the rest of the equipment and processes involved in this operation. Messages will originate on the cockpit data unit (CDU) before being passed to the management unit. (CMU/DMU). In the proposed solution, these messages will then be passed through an encryption layer, using a set encryption key for the message encryption. This encrypted message will then be passed on to the transmission hardware in the same way a traditional message would be.

In reverse, an encrypted message would be received by the transmission hardware, before being passed to the management unit through the encryption layer. In this case, the set key would be used to decrypt the message and pass it to the CDU as per normal operation.

Figure 12 shows a functional block diagram for the proposed solution; however, this solution will need to adhere to a number of restrictions. These restrictions are outlined below.

4.2.1.1 - Radio Transmission Restrictions

Unauthorised transmission of radio messages on a restricted band is prohibited under UK law. This makes it impossible to test the solution on legitimate radio links. To demonstrate the effectiveness of the solution, a radio transmission is substituted by a network link. This allows the messages captured by the solution to be "transmitted" from one entity to another, occurring across a network rather than a radio link.

In addition, the "key exchange" protocol used within the solution is designed as a proof of concept. Should the system be implemented, a suitable manner of transferring encryption keys through out-of-band processes should be determined. This may occur by granting keys to aircrew before boarding, or as part of pre-flight checks.

4.2.1.2 - Formatting and Encoding

Due to the proprietary and restricted nature of the ARINC standards that determines how digital data flows within an aircraft, and the modulation and encoding process used to convert ARINC-standard data into a VDLm2 frame, alternative formatting and encoding is required when capturing and decoding message data.

4.2.2 - Functional Block Breakdown

For simplicity purposes, the stages of the design are broken down into functional blocks. These are discussed below.

4.2.2.1 - Signal Capture

The first functional block in the intended solution is the capture of signals being transmitted from aircraft. This serves two purposes: Evaluating the validity of the solution through the use of legitimate aircraft data and ensuring that any encryption or security protocol is compatible with the data. In addition to this, capturing and successfully decoding live messages will confirm the presence of the security flaws that the project is based around.

4.2.2.2 - Signal Processing

The next functional block revolves around the processing of the signals captured in the previous section. This will cover demodulation, interpretation or decoding, and the output of the raw signals in a compatible format. This section effectively deals with converting the raw signals into individual message frames.

4.2.2.3 - Signal Decoding

The next functional block relates to the decoding of the raw messages from the previous layer. In this section, the content of the raw message frames will be extracted and converted into a format that can be manipulated within a software application.

The result from the capture, processing and decoding blocks should be a collection of messages that can be ingested into the next stage of the application for further processing.

4.2.2.4 - Message Extraction

The next stage of the application will revolve around the extraction of attributes from each individual message. This is to facilitate further sorting and processing of each message, as well as determining which specific system the message is part of. For example, CPDLC messages will have different attributes and fields than ACARS messages. This may make the encryption or protection process different for each different message type.

In addition, the message extraction stage will determine the nature of each message, whether they are strategic, management or diagnostic in nature.

4.2.2.5 - Message Analysis

The analysis stage relates to the analysis of each individual message. At this stage, the messages should be easily addressable.

4.2.2.6 - Message Encryption

This logical block should use some form of encryption method to encrypt and secure the contents of the message. Due to the complexities relating to key exchange and handshake protocols, a symmetric cipher should be employed to streamline the system.

This stage should assign each message a decryption key, for use in the out-of-band key exchange process.

4.2.2.7 - Message Re-Transmission

This next stage will handle the transmission of messages from the "sending" entity to the "receiving" entity, simulating the air-ground link present in the datalink system. This will only involve the encrypted messages, rather than the encryption keys. This is under the assumption that an out-of-band key exchange process is in place that ensures that the ground processing network is in possession of the same keys used on the aircraft.

4.2.2.8 - Message Reception

The message reception block will simulate the receiving ground station network and will ingest encrypted messages from the transmitter. This block will handle the network processes relating to reception of data and store the received data in a suitable data structure for further processing.

4.2.2.9 - Message Decryption

This block will take the encrypted messages from the transmission block and utilise the pre-exchanged keys to convert the encrypted messages back into the original format that was present at message ingestion. The output of this block will be a list of plaintext messages that will then be re-sorted based on their type.

4.2.2.10- Message Display/Output

The final block of the solution will revolve around re-formatting the messages into the original format they arrived in. This is to highlight the transparent nature of the layer. Theoretically, this will mean that the message from the CDU will arrive at the layer in the ARINC 429 format, and leave the encryption layer on the ground network, after having been encrypted, transmitted, and decrypted, in the same format that a non-encrypted message would arrive in. This is to ensure the compatibility between systems.

In addition, the re-formatting of the messages will also allow a comparison to be made between the input and output. This will determine the potential for message loss.

4.2.2.11- Key Exchange Process

The key exchange process is a critical aspect of the proposed solution. Whilst this is not a functional block in of itself, the process of transmitting and exchanging encryption keys is critically important to the correct functioning of the solution.

An asymmetric cipher would require significant key exchange infrastructure as part of the design. This is also reflected in the literature review, with key management being a significant hurdle in the field of aeronautical datalinks, as well as in other industries.

As a result of the difficulties identified with asymmetric encryption, the solution will use a symmetric cipher to provide encryption and decryption functionality. Whilst there are significant shortcomings with this, such as the potential loss of security should the encryption key be disclosed, designing, and implementing key exchange as part of the solution is beyond the scope of the project.

It is therefore assumed that key exchange would occur through an out-of-band transmission method. That is, either before a flight departs or establishes datalink connection for the first time, the aircrew would be granted the specific key for use in flight. This key would already have been shared amongst the ground station network beforehand, through the standard networks used as part of their infrastructure.

The solution therefore assumes that whilst in flight, the ground station network and airborne entities are in possession of the same symmetric cipher key.

4.2.3 - Objectives List

This section contains a list of objectives that the designed solution should meet. This can be used in the testing and evaluation sections to determine the effectiveness of the overall solution at meeting the requirements set by the problem domain.

#	Objective	Weight
0	Receive and capture datalink messages	MUST
1	Decode datalink messages	MUST
2	Output readable files for further processing	MUST
3	Output files in a well known format	SHOULD
4	Sort and categorise messages for further analysis	COULD
5	Allow the viewing, management and analysis of messages	COULD
6	Use a suitable cipher to encode the messages	MUST
7	Use a suitable key management method to transfer the keys between entities	SHOULD
8	Use a suitable transmission medium to transfer messages	MUST
9	Messages should be able to be transferred bidirectionally	COULD
10	Messages should be received correctly on the remote instance	MUST
11	Messages should be decrypted using the correct key	MUST
12	Intercepted messages should not be able to be decoded without the key	MUST

5. Implementation and Testing

5.1 - Signal Capture, Processing, and Decoding

The first three logical blocks revolve around the capture of signals, the processing and demodulation of these signals, and the eventual decoding of these signals into interpretable messages. The reason for these blocks is two-fold:

1. Verification of security flaws within the network and viability of potential traffic capture
2. Providing data for the artifact to use to increase viability of proposed system.

To complete this stage, several hurdles needed to be overcome. These relate to the availability of the raw VDLm2 messages in the geographical area, the ability to distinguish these messages from the local noise floor (the amount of radio interference in the area), and to ensure that any messages distinguished would have a high enough signal-to-noise ratio to facilitate data extraction.

5.1.1 - Message Availability

VHF ACARS messages, transmitted using VDLm2 datalinks, have a range of approximately 200 nautical miles. (British Aerospace, 1997). The expected range for a VDLm2 message is highly dependent on the altitude of the transmitting aircraft, and the altitude of the receiving ground station. VDLm2 messages utilise the VHF radio band, limiting the transmissions to line-of-sight.

In order to determine whether there was any potential for message capture, analysis using a suitable radio receiver was required.

In traditional radio hardware, signal processing is carried out in physical circuits. This complexity makes these devices expensive and difficult to obtain. Software defined radio (SDR) devices utilise the power of modern computers to carry out the previously hardware-based signal processing in software. This greatly reduces the cost and permits the development of software applications to decode and interpret signals.

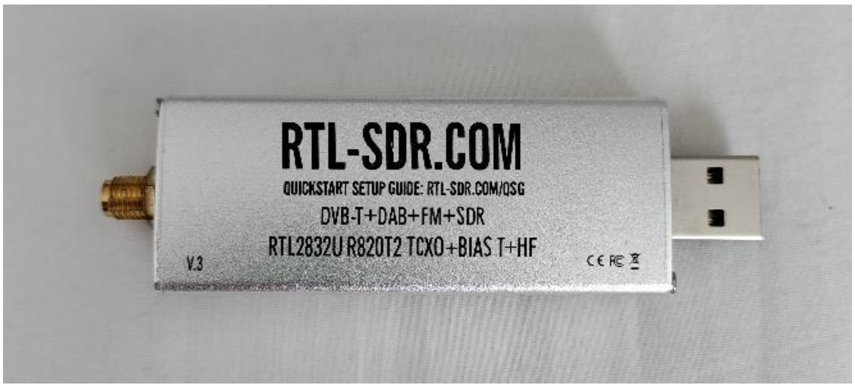

RTL-SDR.COM
QUICKSTART SETUP GUIDE: RTL-SDR.COM/QSG
DVB-T+DAB+FM+SDR
RTL2832U R820T2 TCXO+BIAS T+HF
V.3

Figure 14 - RTL-SDR Device used

To perform analysis and message capture, a simple SDR was used - An RTL2832U based receiver, obtained from www.RTL-SDR.com; a well-known provider of SDR equipment. These devices are common within the SDR landscape. Originally based on modified TV-tuner devices, these are capable of tuning to large swathes of the radio spectrum, including the aeronautical bands. (Robert W. Stewart, 2015).

RTL-SDR devices such as these are increasingly used in development and test environments, as well as within the world of amateur radio, due to their low cost and ease of use.

For analysis, the receiver is located at approximately 10m above sea level, with a good view of the sky in all directions. In addition to this, the receiver is within 10 miles of a VHF omnidirectional range (VOR) navigation beacon. This makes it ideally suited to capture air traffic data.

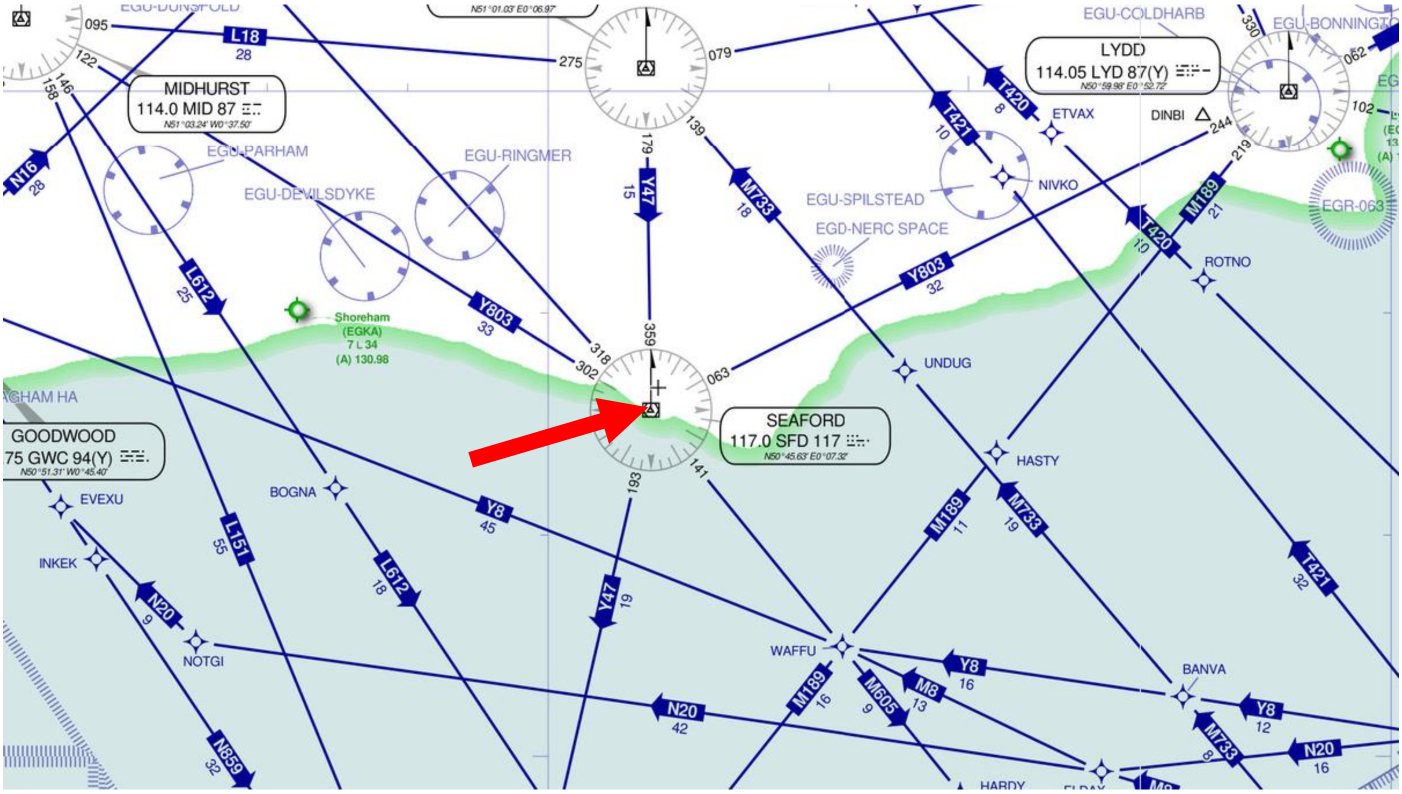


Figure 15 - Section of low-level IFR Chart highlighting the location of the receiver in relation to common airways. (Skyvector, n.d.)

A key consideration when attempting to interpret signals is the choice of antenna. VDLm2 datalinks are transmitted in the 118-136Mhz aviation band. The formula for calculating the required antenna length is as follows:

$$F = \frac{C}{\lambda}$$

Where F is the frequency in hertz, C is the speed of light in a vacuum, and λ is the wavelength of the signal. For VDLm2 messages, the primary frequency in use is 136.957Mhz, equating to an antenna/wavelength of approximately 220cm. A standard, end-fed scanner antenna was used, with a frequency range of 136-145Mhz. This provided adequate performance in the relevant band.

According to a table published by the ICAO, detailing the use of frequencies within the aeronautical band, a number of frequencies are allocated for VDLm2 based datalinks in the general area of the receiver:

Location	Frequency	Type	Desc
London/City	131.725	DL	ACARS
	131.825	DL	ACARS, ARINC
	136.975	DL	VDL; M2; SITA
	136.725	DL	VDL; M2; ARINC
	136.975	DL	VDL; M2; ARINC
	136.825	DL	VDL; M2; ARINC
London/Gatwick	131.825	DL	ACARS, ARINC
	136.975	DL	VDL; M2; ARINC
	136.975	DL	VDL; M2; SITA
	136.725	DL	VDL; M2; ARIC; SITE A
	131.725	DL	ACARS, SITA
	131.525	DL	ACARS; SITA; LGW3

Table 4 - Extract of ICAO Doc 7754 (ICAO, 2022)

Of these frequencies, the one with the best performance and overall best availability is 136.975Mhz, which is utilised for VDLm2 messages from SITA, a provider for information technology services to the aviation industry.

The next step is to confirm availability of messages within the area of the receiver. In order to do this, another piece of software is required to inspect the radio spectrum in the area of the receiver.

5.1.2 – SDR#

SDR# is a general-purpose SDR application, designed to inspect the radio spectrum and perform general monitoring. SDR# is compatible with a number of different sources, including a physical RTL-SDR devices such as the one in Figure 14. SDR# can also be augmented with plugins that allow the decoding of common radio communication protocols, such as TETRA, DMR, and METEOR weather satellites.

Figure 16 shows a VDLm2 message, as displayed within SDR#. The upper display is a spectrum analyser, showing peaks in areas of high signal. The lower display is a waterfall diagram, where past signals appear as bright lines, allowing the historical viewing of signals within a specific band. The vertical lines present in the waterfall display and the associated peaks indicate sources of strong background noise or interference.

The area highlighted with the red box is a VDLm2 message that was transmitted within the range of the receiver. VDLm2 messages are transmitted as bursts, lasting a fraction of a second each.

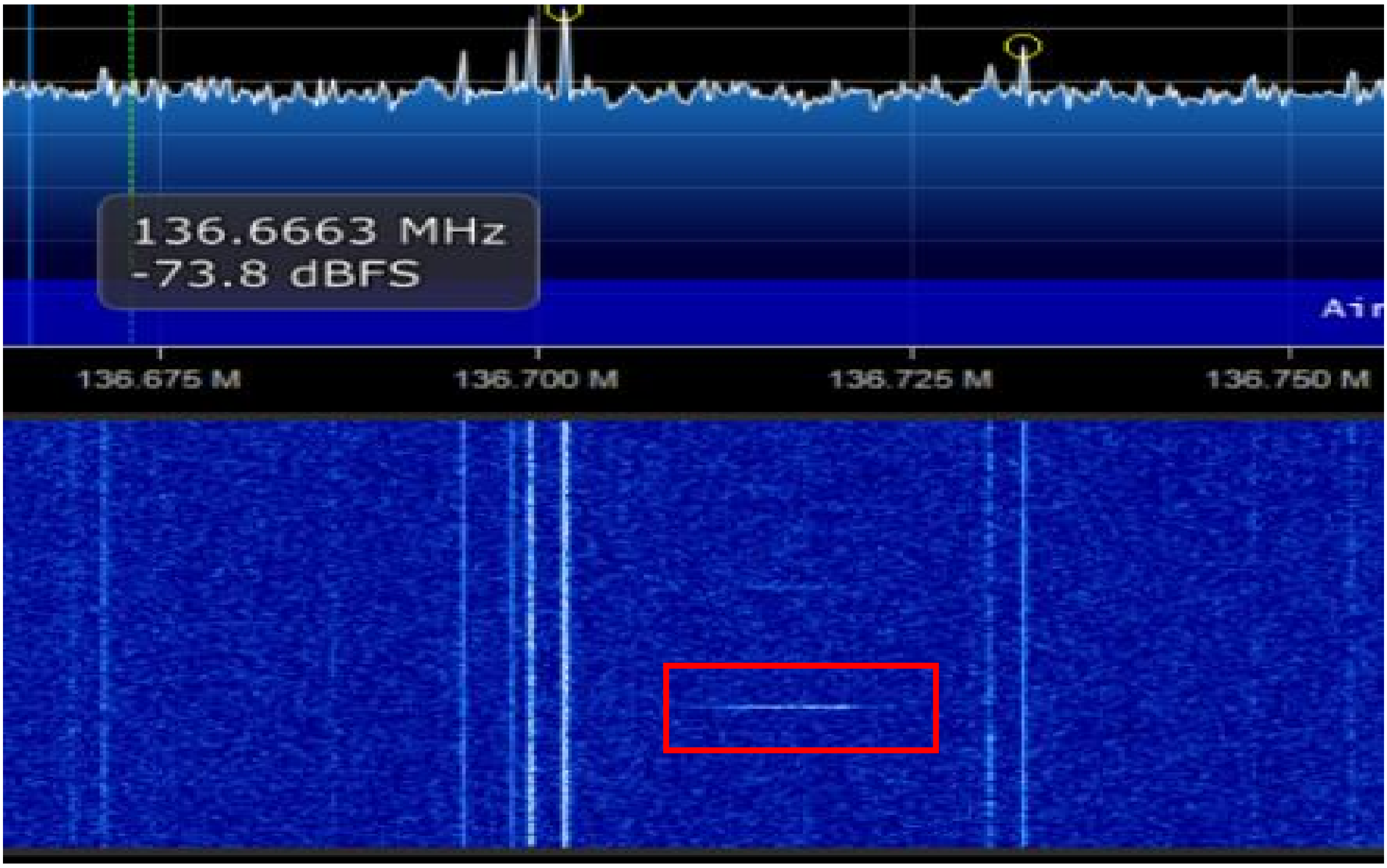


Figure 16 - Screenshot showing VDLm2 message

SDR# played a critical element in tuning and identifying the frequencies most in use in the location of the transmitter. Due to hardware design, there is also a slight frequency offset. This can be identified within SDR# and used in the decoding stage of the artefact, to better increase interception performance.

5.2 - Interpretation and Decoding

Since the verification and capture stages are complete, the next section is to decode the messages that are present. Due to the complexity around this task, and the fact that the goal of the project is to implement a security layer rather than decode the messages, a third-party application was used to interpret these messages.

5.2.1 - Software Overview

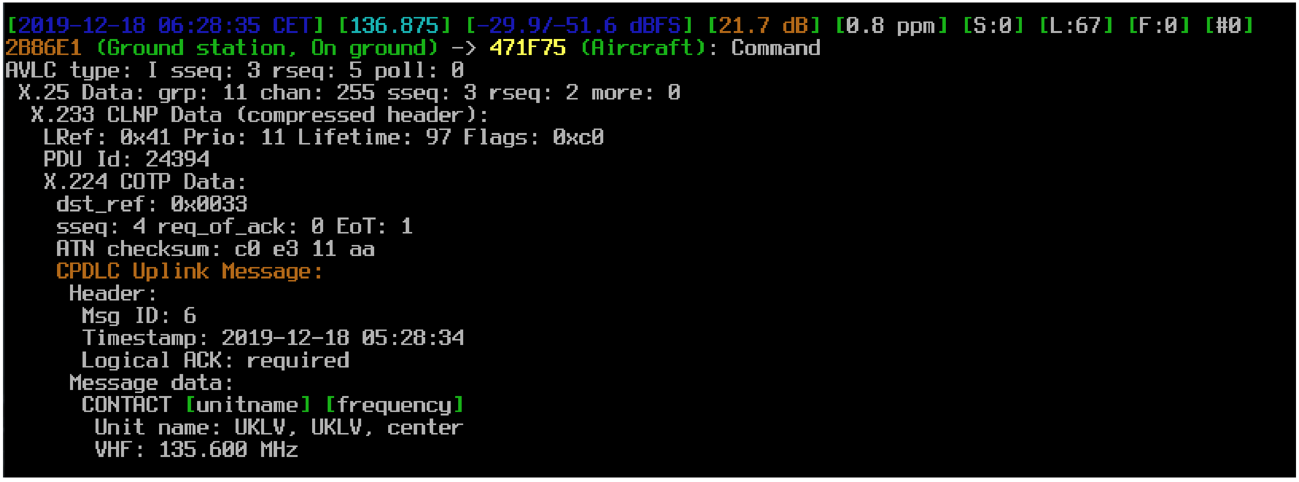
A number of software implementations exist that specialise in decoding intercepted VDLm2 messages. These are shown in the table below.

#	Name	Desc
0	ACARSd	ACARSd attempts to decode ACARS messages intercepted by an aviation-band radio, through a computer's sound card. (acarsd, 2022)
1	MultiPSK	MultiPSK is a freeware application that supports a number of low-level radio protocols, commonly utilised in amateur radio, or "ham" equipment.
2	DumpVDL2	DumpVDL2 is a "VDL Mode 2 Message Decoder and Protocol Analyzer", released under the open-source GPL3.0 licence. (Lemiech, 2021). DumpVDL2 interfaces with the RTL-SDR device, and supports the reception and decoding of a number of VDLm2 formats, including ACARS over AVLC and ATN-B1 CPDLC.

One of the primary features required when identifying the right decoding software to use is the ability to utilise the messages output by the program in another application. For this to be possible, the messages must be output in an easily readable format.

Overall, the program with the most success in decoding messages captured by the RTL-SDR device is DumpVDL2. ACARSd does not have support for the required hardware, instead only supporting an Airband radio connected through the sound card. MultiPSK is primarily designed for use with low-level amateur radio protocols, and therefore is not as suited to the application when compared to others.

DumpVDL2 was therefore chosen to interpret the intercepted messages from the RTL-SDR dongle.



```
[2019-12-18 06:28:35 CET] [136.875] [-29.9/-51.6 dBFS] [21.7 dB] [0.8 ppm] [S:0] [L:67] [F:0] [#0]
2B86E1 (Ground station, On ground) -> 471F75 (Aircraft): Command
AVLC type: I sseq: 3 rseq: 5 poll: 0
 X.25 Data: grp: 11 chan: 255 sseq: 3 rseq: 2 more: 0
  X.233 CLNP Data (compressed header):
   LRef: 0x41 Prio: 11 Lifetime: 97 Flags: 0xc0
   PDU Id: 24394
   X.224 COTP Data:
    dst_ref: 0x0033
    sseq: 4 req_of_ack: 0 EoT: 1
    ATN checksum: c0 e3 11 aa
    CPDLC Uplink Message:
     Header:
      Msg ID: 6
      Timestamp: 2019-12-18 05:28:34
      Logical ACK: required
     Message data:
      CONTACT [unitname] [frequency]
       Unit name: UKLV, UKLV, center
       VHF: 135.600 MHz
```

Figure 17 - Example DumpVDL2 Output (Lemiech, 2021)

DumpVDL2 can be executed from the command line, with the following syntax:

```
dumpvdl2 -output decoded:json:file:path=/FILEPATH,rotate=hourly --rtlsdr0 --
gain 40 136725000 136975000 136875000
```

The first argument "—output" specifies the output of the decoder. For the artifact, the decoder outputs the decoded messages in JSON format, at the specified file path. The output files consist of multiple JSON objects, and will rotate on an hourly basis to ensure stability in the event of a crash, and to ensure consistent file sizes.

The next argument, "—rtlsdr 0" indicates the specific receiver to use. This can be useful when multiple receivers are installed. In the case of this setup, only one is present.

Next, the gain is supplied. Too high a gain setting will result in message distortion, too low will result in too little fidelity in the messages, and the decoder will fail to interpret them. A medium-high value of 40dB is used.

5.2.2 - Message Decoding

DumpVDL2 was compiled, installed and run on the receiver at the location specified earlier in the chapter, on four consecutive occasions. A breakdown of the data collected is given below.

Table 5 - Breakdown of message capture

Date	Start Time	End Time	Messages Captured
21/02/2022	09:00	09:59	*3292*
	10:00	10:59	*2663*
	11:00	11:59	*2718*
	12:00	12:59	*2570*
	13:00	13:59	*1221*

22/02/2022	12:00	12:59	*2169*
	13:00	13:59	*2260*
	14:00	14:59	*2281*
	15:00	15:59	*467*
23/02/2022	00:00	00:59	*86*
	01:00	01:59	*67*
	10:00	10:59	*475*
	11:00	11:59	*2075*
	12:00	12:59	*2259*
	13:00	13:59	*1095*
Total Monitoring Hours		**14**	***25,698***

Table 5 shows the data collected after monitoring 136.975Mhz across three separate days. Gaps in the monitoring were due to external constraints on the hardware. In total, over twenty-five thousand messages were received over fourteen hours of monitoring.

Of the messages that were received, a further breakdown can be performed:

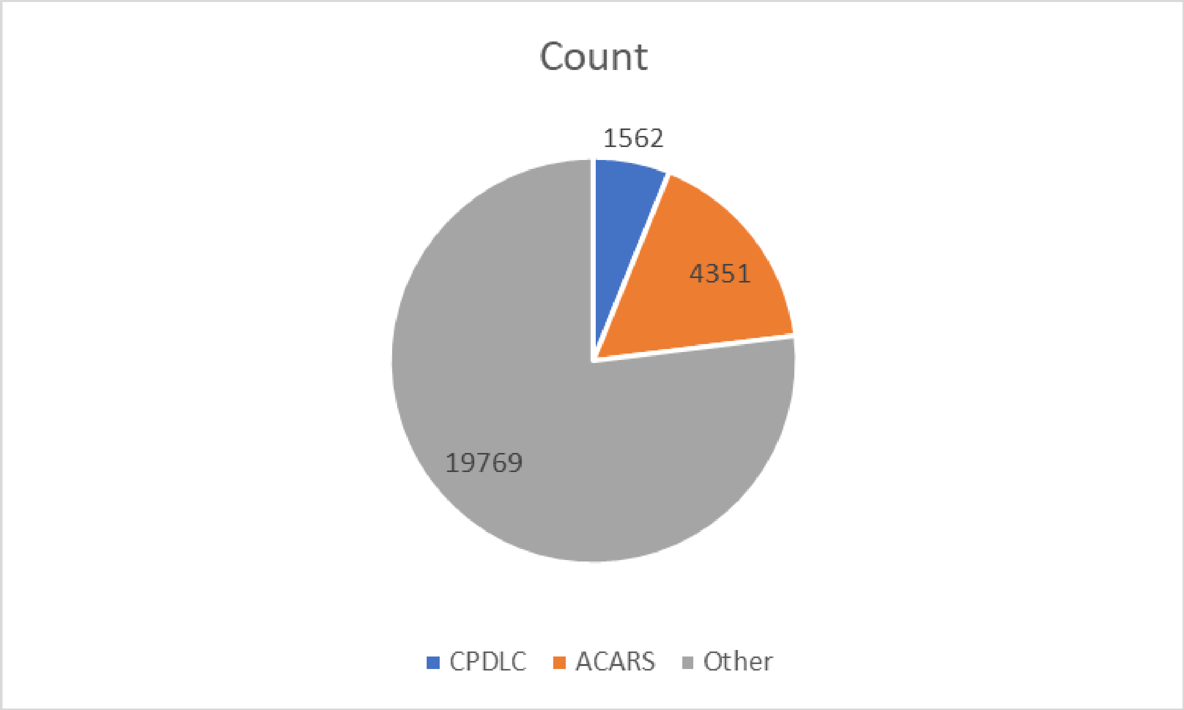
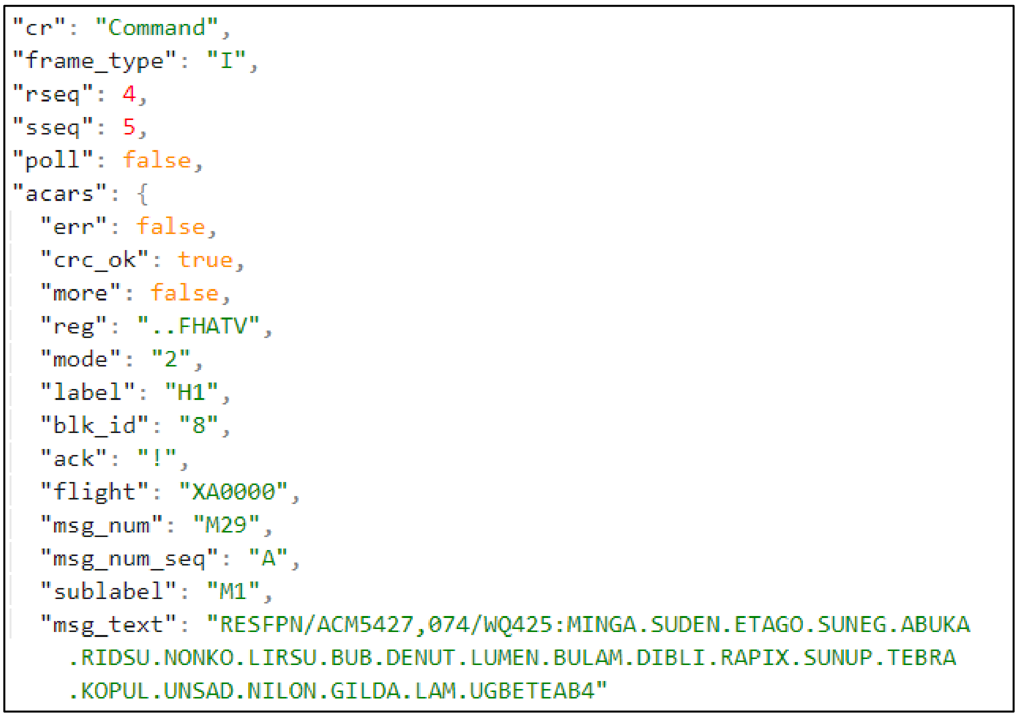


Figure 18 - Chart showing type of message throughout the captured data

Figure 18 shows the breakdown of message types throughout the captured data. A high proportion of "other" message categories is present. These messages do not fall under CPDLC or ACARS systems and may be part of other infrastructure that uses VDLm2 as a messaging protocol. Documentation regarding these messages is relatively limited and beyond the scope of the project.

5.2.2.1 - Examples of Captured messages

The messages captured by DumpVDL2 consist of several different subtypes. This gives an insight into the types of system in operation in the area, as well as the nature of some of these messages. Some of the different types of messages are illustrated below.



```
"cr": "Command",
"frame_type": "I",
"rseq": 4,
"sseq": 5,
"poll": false,
"acars": {
  "err": false,
  "crc_ok": true,
  "more": false,
  "reg": "..FHATV",
  "mode": "2",
  "label": "H1",
  "blk_id": "8",
  "ack": "!",
  "flight": "XA0000",
  "msg_num": "M29",
  "msg_num_seq": "A",
  "sublabel": "M1",
  "msg_text": "RESFPN/ACM5427,074/WQ425:MINGA.SUDEN.ETAGO.SUNEG.ABUKA
    .RIDSU.NONKO.LIRSU.BUB.DENUT.LUMEN.BULAM.DIBLI.RAPIX.SUNUP.TEBRA
    .KOPUL.UNSAD.NILON.GILDA.LAM.UGBETEAB4"
```

Figure 19 - Example of ACARS message captured

Figure 19 shows a collection of the attributes of a captured ACARS message. The content of the message appears to show a period-separated list of waypoints. Passing these waypoints into flight planning software allows a route to be generated.

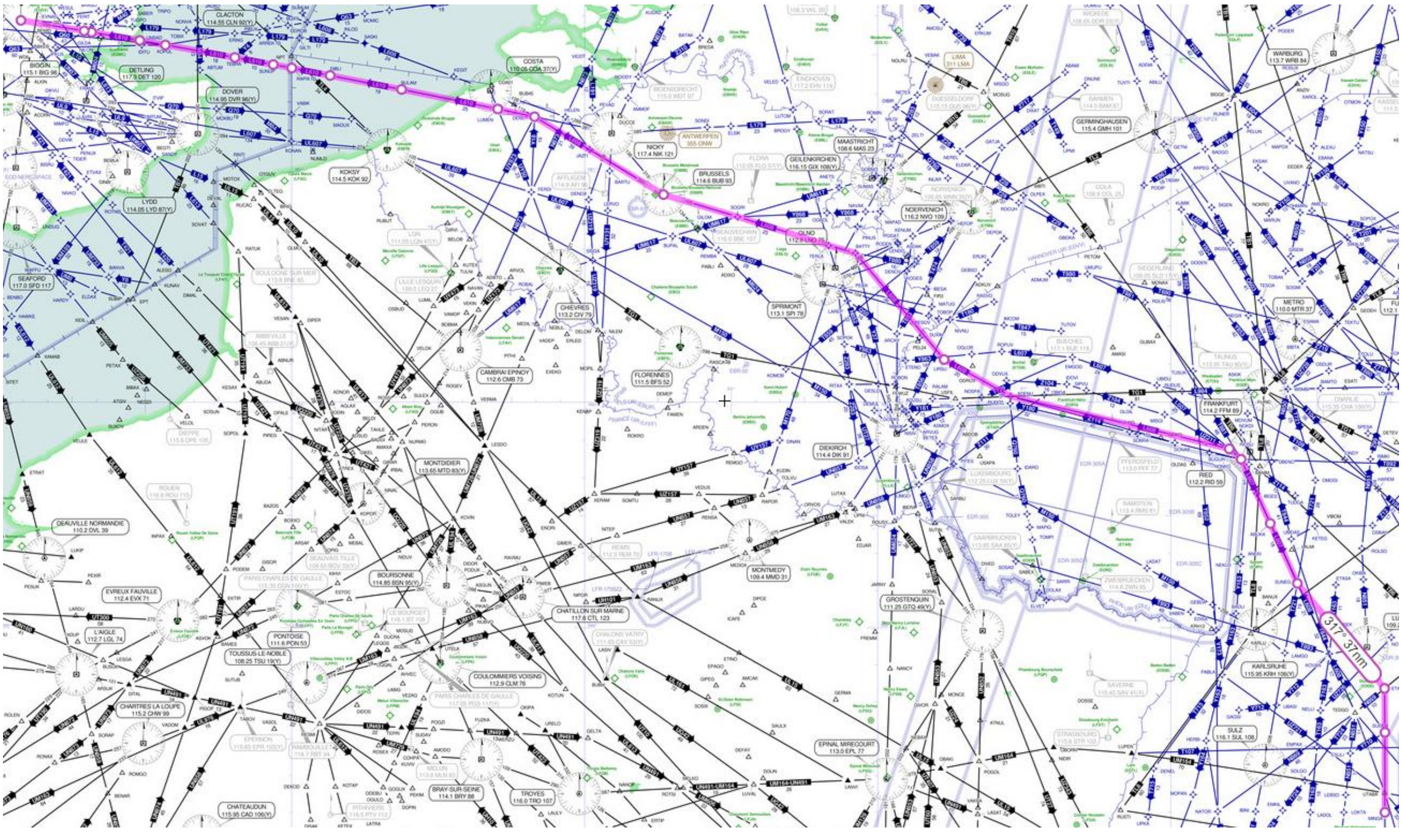
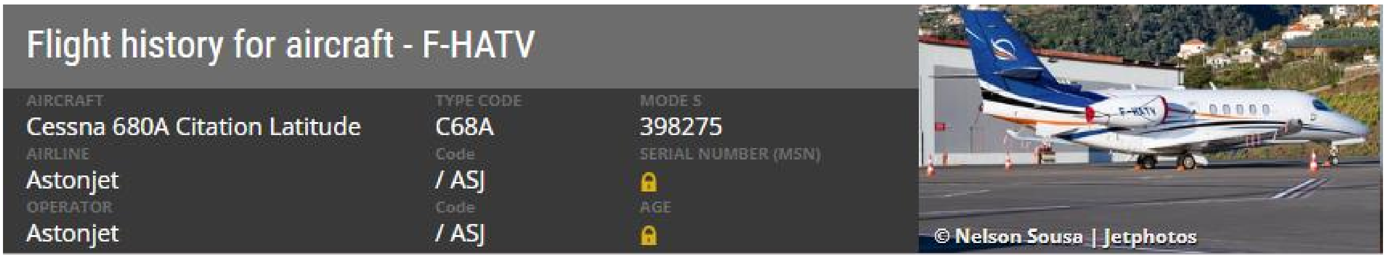


Figure 20 - Generated flight plan using ACARS message content (Skyvector, n.d.)

Figure 20 shows a route generated using the contents of an ACARS message captured using DumpVDL2. Along with the message text, additional attributes are also given. These attributes show the registration of the aircraft, the assigned flight number and the 24-bit ICAO Aircraft address (not shown in fig.20). These details can then be used to search for the aircraft itself, in a commercial flight tracking website such as Flightradar24.



Flight history for aircraft - F-HATV

AIRCRAFT
Cessna 680A Citation Latitude

TYPE CODE
C68A

MODE S
398275

AIRLINE
Astonjet

Code
/ ASJ

SERIAL NUMBER (MSN)

OPERATOR
Astonjet

Code
/ ASJ

AGE

© Nelson Sousa | Jetphotos

Figure 21 - Aircraft search results after searching using the aircraft tail number (F-HATV) (FlightRadar24, 2022)

To summarise, a message captured using the RTL-SDR device and an open-source decoder (DumpVDL2) has allowed the flight plan for an aircraft to be determined. The aircraft is registered to an airline called AstonJet, which specialises in private charter flights. (AstonJet, n.d.).

The presence of this information and its ease of access verifies the potential for security risks. This also verifies that data can be captured from this system and decoded successfully, resulting in a usable dataset for the next element of the project.

5.3 - Message Extraction

5.3.1 – Artefact Introduction and General Overview

So far, third party software has been used to carry out required operations that are beyond the scope of the project, such as signal demodulation and message decoding. The result is a collection of JSON message objects, output by the decoding software, 'DumpVDL2'.

The artefact is split into two sections. A receiver, and a transmitter. The main reason for this is to simulate an aircraft entity (the transmitter) and a ground reception entity (the receiver). In a traditional datalink system, messages are sent bi-directionally between the two entities. The solution revolves around the implementation of a 'secure layer' within this system, to protect message contents. In terms of the artefact, the 'receiver' (ground network) will listen for encrypted messages from the 'transmitter' (airborne entity).

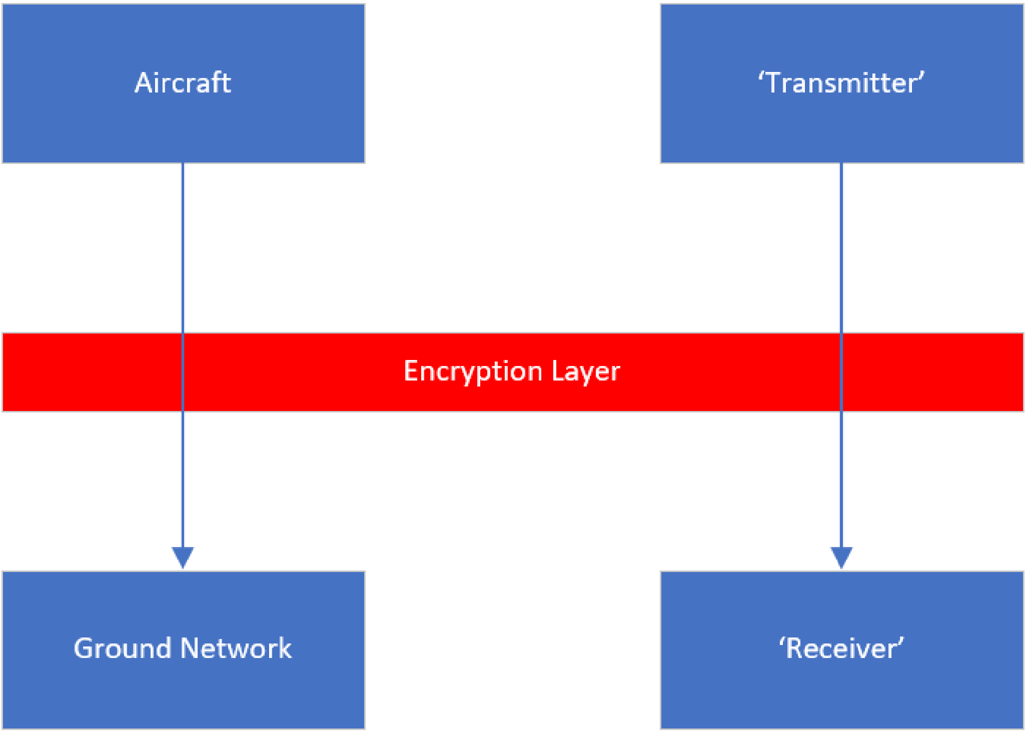


Figure 22 - Abstract overview of artefact against a traditional datalink system

Figure 22 highlights the relationship between a traditional datalink network and the one simulated within the artefact. The use of the two copies of the artefact running at the same time to simulate the system will allow the secure layer to be demonstrated and assessed.

The next functional block revolves around extracting the required information from the message. The previous sections have focused on obtaining and capturing the messages, which is now complete. The next stage of the solution will utilise this captured data in the encryption and transmission layers. First, the captured data will be analysed, with the attributes of each message extracted and stored in a suitable data structure. Once this is complete, the messages will be converted into an encrypted format, before being finally transmitted to another instance of the software, where they will undergo decryption.

The message output from DumpVDL2 is a list of JSON-formatted objects. Each one of these objects is read by the artefact, and the attributes extracted. Python was used as the main codebase for the artefact, due to its ease of use, speed, and easy-to-read syntax.

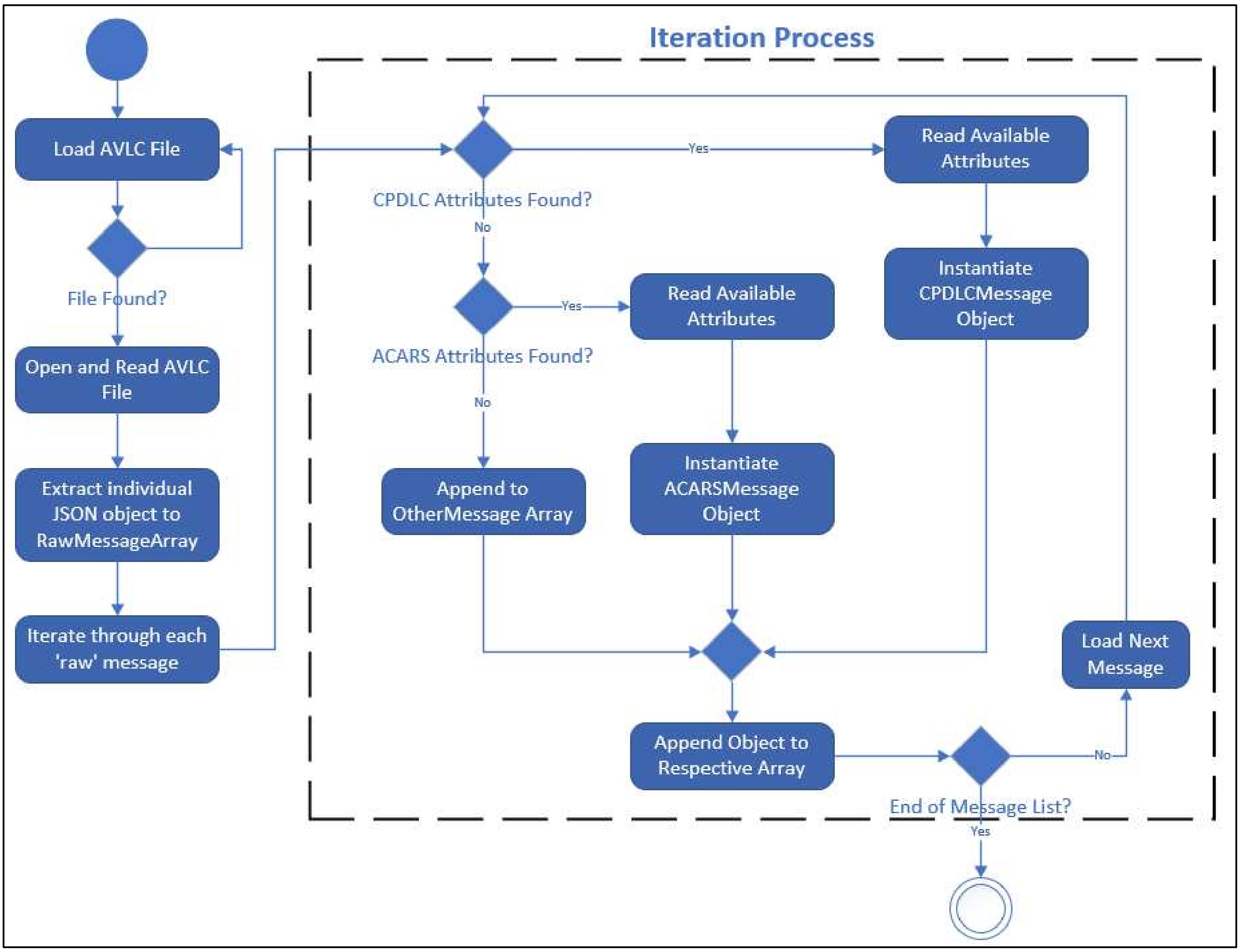

Iteration Process
Load AVLC File
File Found?
Open and Read AVLC File
Extract individual JSON object to RawMessageArray
Iterate through each 'raw' message
CPDLC Attributes Found?
Yes
No
Read Available Attributes
Instantiate CPDLCMessage Object
ACARS Attributes Found?
Yes
No
Read Available Attributes
Instantiate ACARSMessage Object
Append to OtherMessage Array
Append Object to Respective Array
End of Message List?
No
Yes
Load Next Message

Figure 23 - ExtractMessagesFromFile() Function activity diagram

The output from DumpVDL2 is renamed with a '*.avlc' extension, to differentiate it from other JSON files and to highlight them in the .gitignore file. This file determines what files are excluded from the version control system, therefore preventing them from being uploaded to Github. This helps to ensure message privacy.

Figure 23 shows an activity diagram for the message extraction process used within the artefact. An AVLC file is loaded, and each JSON object within the file is assigned to an index within *RawMessageArray*. This array is then passed to the *ExtractMessagesFromFile()* function, whereupon the function iterates through every message within the array, to determine the attributes present. If attributes belonging to an ACARS message are found, the contents of the message are used to instantiate an *ACARSMessage* object. This *ACARSMessage* object is then appended to the *ACARSMessageObjects* array at the end of the iteration, for use later in the program.

The nature of CPDLC messages mean that not all message attributes are present in every message. This makes instantiating a *CPDLCMessage* object difficult, as it is not known what attributes are present before instantiation. For this reason, if one CPDLC attribute is found, all other potential attributes are checked for. If they are found, they are used to instantiate an object. If they are not found, "null" will be passed as the value of that attribute instead. This ensures all required attributes are passed to the object at instantiation.

If no attributes belonging to CPDLC or ACARS messages are found, the contents of the message are then appended to an "other" message array. Once this has been completed, the function will then complete the same process for the next raw message in the array, until the entire array has been iterated through.

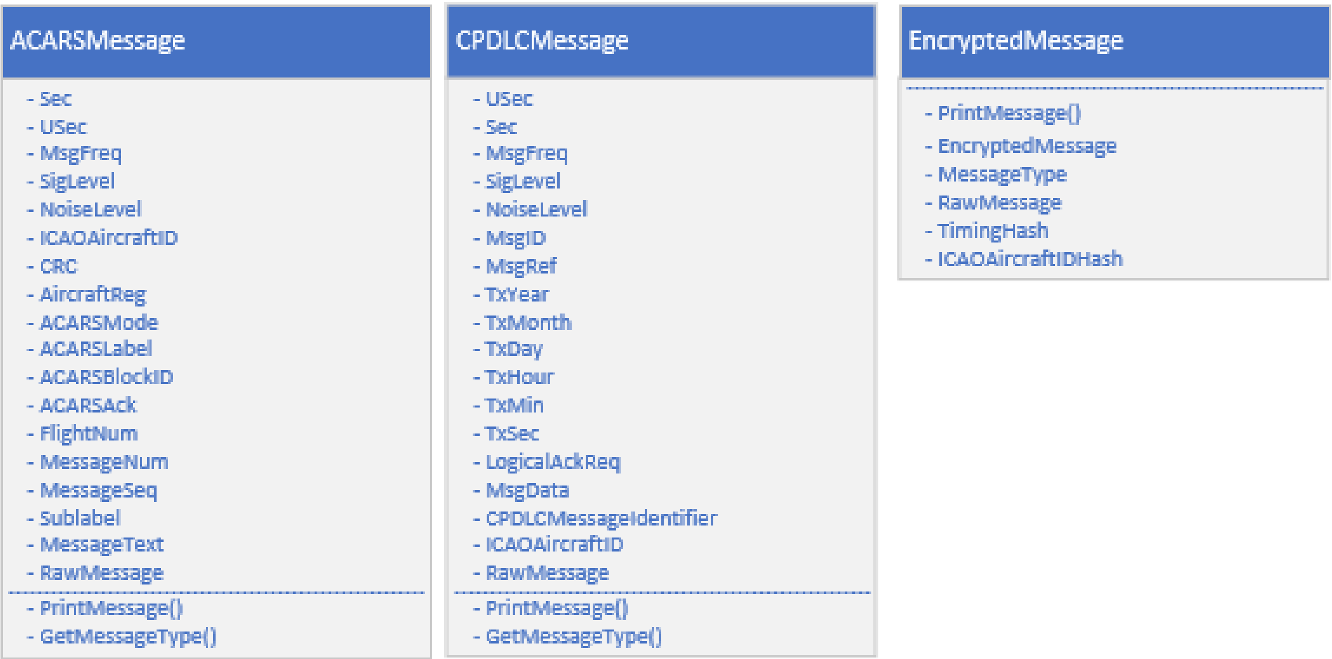
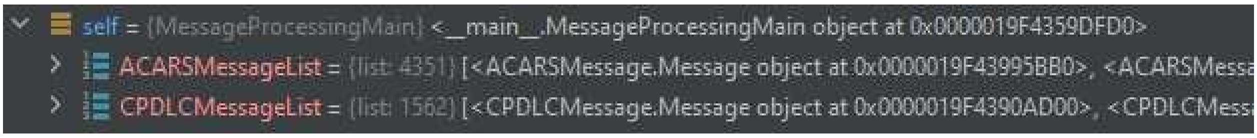


Figure 24 - UML Class Diagrams for message data structures

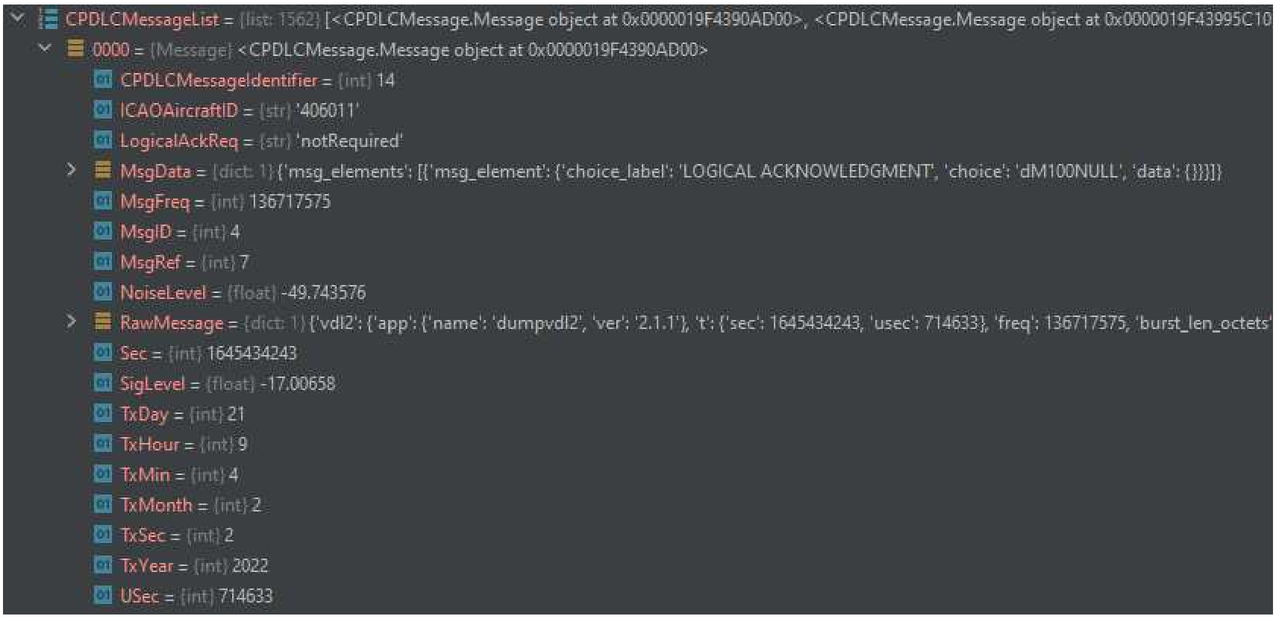
At the end of this extraction block, each message will be stored in its associated array. The format of each message is given above, with class diagrams for each type of message shown in Figure 24. The content of each message is used to instantiate the associated object, which is then stored within an array for each message type.



```
v  self = {MessageProcessingMain} <__main__.MessageProcessingMain object at 0x0000019F4359DFD0>
  >  ACARSMessageList = {list: 4351} [<ACARSMessage.Message object at 0x0000019F43995BB0>, <ACARSMessa
  >  CPDLCMessageList = {list: 1562} [<CPDLCMessage.Message object at 0x0000019F4390AD00>, <CPDLCMess
```

Figure 25 - Python Debugger output showing stored array objects

Figure 26 shows the output of the python built-in debugger, during program execution. This shows stored ACARSMessage and CPDLCMessage objects in their associated arrays.



```
v  CPDLCMessageList = {list: 1562} [<CPDLCMessage.Message object at 0x0000019F4390AD00>, <CPDLCMessage.Message object at 0x0000019F43995C10
  v  0000 = {Message} <CPDLCMessage.Message object at 0x0000019F4390AD00>
       CPDLCMessageIdentifier = {int} 14
       ICAOAircraftID = {str} '406011'
       LogicalAckReq = {str} 'notRequired'
    >  MsgData = {dict: 1} {'msg_elements': [{'msg_element': {'choice_label': 'LOGICAL ACKNOWLEDGMENT', 'choice': 'dM100NULL', 'data': {}}}]}
       MsgFreq = {int} 136717575
       MsgID = {int} 4
       MsgRef = {int} 7
       NoiseLevel = {float} -49.743576
    >  RawMessage = {dict: 1} {'vdl2': {'app': {'name': 'dumpvdl2', 'ver': '2.1.1'}, 't': {'sec': 1645434243, 'usec': 714633}, 'freq': 136717575, 'burst_len_octets'
       Sec = {int} 1645434243
       SigLevel = {float} -17.00658
       TxDay = {int} 21
       TxHour = {int} 9
       TxMin = {int} 4
       TxMonth = {int} 2
       TxSec = {int} 2
       TxYear = {int} 2022
       USec = {int} 714633
```

Figure 26 - CPDLCMessage object showing correct attributes

By drilling down into the contents of each of the arrays and the stored objects within, Figure 26 shows a CPDLCMessage object with attributes, highlighting the correct sorting and allocation of data throughout the function.

5.4 - Message Analysis

The next logical block relates to message analysis. This is supplemental to the artefact itself,

5.5 - Message Encryption

This logical block relates to the encryption of each message object itself. This is a critically important element in the overall solution, as the strength and security of the implementation is determined here.

The encryption process of the messages has a number of challenges to overcome. For example, the key management process, encryption type, and cipher to use are all elements that have a number of problems.

Key Management Process

As mentioned before, the key management process is a critical issue in regards to aeronautical datalinks. To solve this problem, an out-of-band key exchange method is used. This is beyond the scope of the report, and it is assumed that there will be another means of transferring keys between ground entities and aircraft, before the flight is underway.

In the case of the artefact, a network transmission of encrypted messages is used to simulate the datalink, and the storage and retrieval of keys from a shared file is used as an implementation of the "out-of-band" transmission protocol used for key distribution.

Encryption Type and cipher

For encryption, the solution uses the Fernet cipher, a python implementation of the AES-128 algorithm, in CBC mode. According to the documentation (tmaher, 2014), Fernet takes a 'user message' in the form of a sequence of bytes, a 'key', and uses this information and the current time to create a 'token'. The token is the encrypted message, which is inaccessible without the key.

	Fernet Token Format				
Field Name	*Version*	*Timestamp*	*Initialization Vector*	*Ciphertext*	*HMAC*
Field Size (b)	*8*	*64*	*128*	*N*128*	*256*

Table 6 - Fernet Token Format (tmaher, 2014)

Table 6 shows the layout of a Fernet token, along with field size.

Fernet as a method to encrypt and secure previously insecure communication protocols has been attempted before, with a paper (El Gaabouri Ismail, 2020). The paper highlights one of the key advantages of Fernet encryption, that being its lightweight footprint and ability to operate on relatively limited hardware. For this reason, Fernet encryption was used as the primary cryptographic operative in the solution, with the exact implementation detailed below.

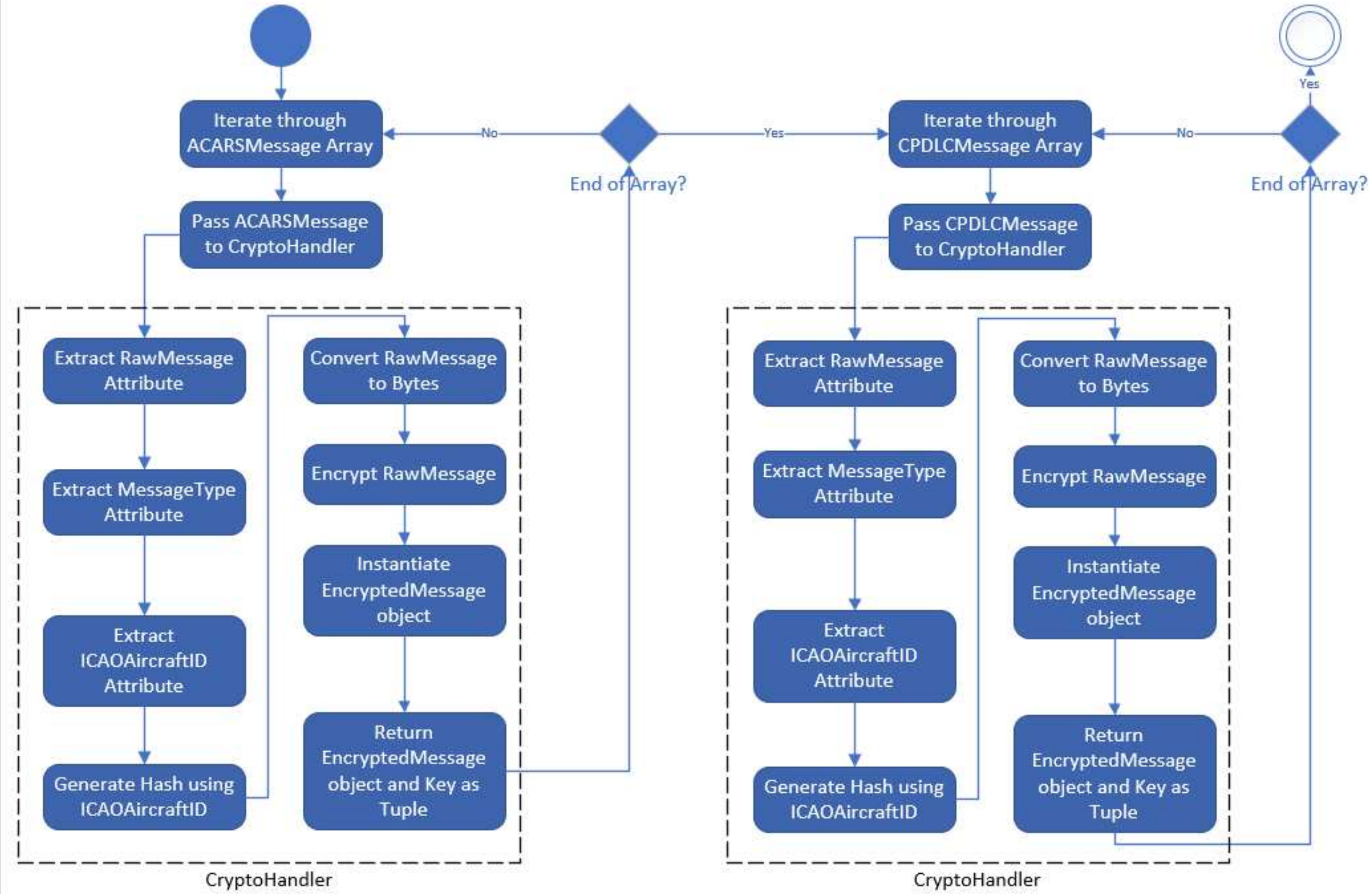
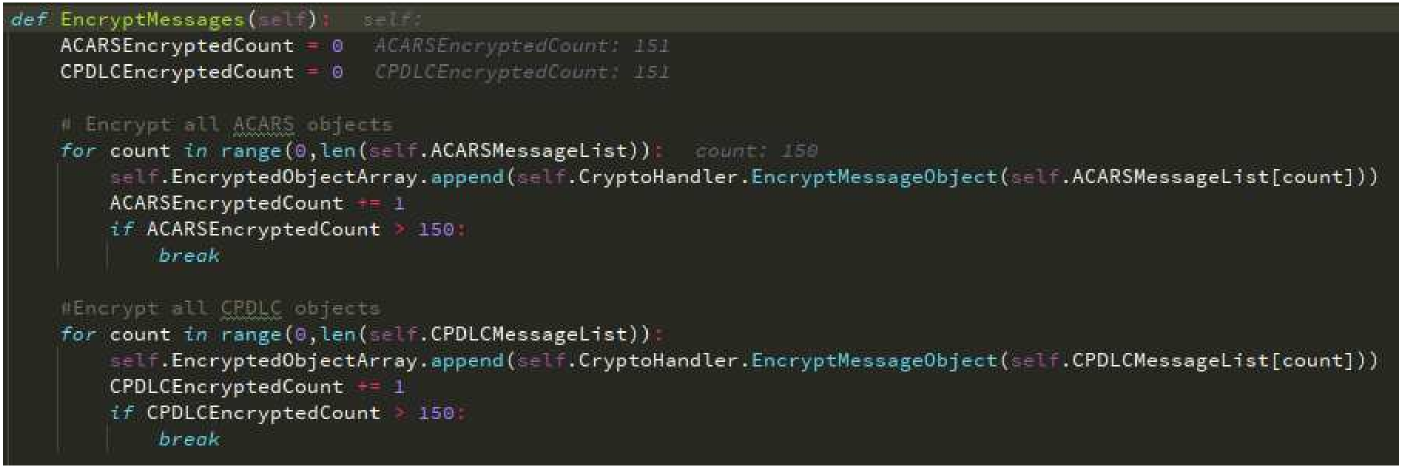

Iterate through ACARSMessage Array
No
End of Array?
Yes
Iterate through CPDLCMessage Array
No
End of Array?
Yes
Pass ACARSMessage to CryptoHandler
Pass CPDLCMessage to CryptoHandler
Extract RawMessage Attribute
Extract MessageType Attribute
Extract ICAOAircraftID Attribute
Generate Hash using ICAOAircraftID
Convert RawMessage to Bytes
Encrypt RawMessage
Instantiate EncryptedMessage object
Return EncryptedMessage object and Key as Tuple
CryptoHandler
Extract RawMessage Attribute
Extract MessageType Attribute
Extract ICAOAircraftID Attribute
Generate Hash using ICAOAircraftID
Convert RawMessage to Bytes
Encrypt RawMessage
Instantiate EncryptedMessage object
Return EncryptedMessage object and Key as Tuple
CryptoHandler

Figure 27 - Overview of Encryption Process

The 'transmitting' entity will run the encryption process after ingesting raw message data from the decoder. The encrypted messages will then be transmitted to the remote 'receiver' for decoding. Figure 28 shows the process in which the transmitter encrypts the sorted messages. First, the ACARSMessage array is iterated through, and key attributes from the messages are extracted. The 'RawMessage' attribute is used as the main ciphertext of each encrypted message and is a copy of the original JSON message. The 'MessageType' attribute is stored in the 'EncryptedMessage' object along with the ciphertext to aid in message identification before it is transmitted. The 24-bit ICAO Aircraft ID is then extracted and used to identify the aircraft that encrypted the message.

Once these attributes have been extracted, the raw message is encoded into a byte string, whereupon it is then passed to the Fernet encryption module and encrypted. On the transmitter side, the associated message key is stored with the EncryptedMessage object, for writing to the key file.



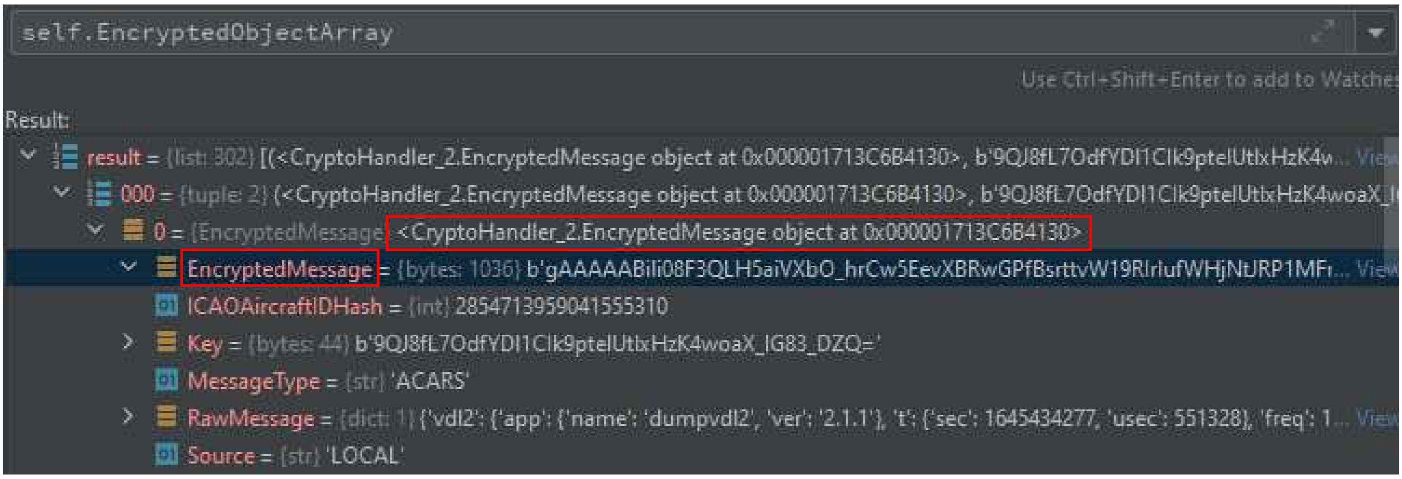
```
def EncryptMessages(self):   self:
    ACARSEncryptedCount = 0   ACARSEncryptedCount: 151
    CPDLCEncryptedCount = 0   CPDLCEncryptedCount: 151

    # Encrypt all ACARS objects
    for count in range(0,len(self.ACARSMessageList)):   count: 150
        self.EncryptedObjectArray.append(self.CryptoHandler.EncryptMessageObject(self.ACARSMessageList[count]))
        ACARSEncryptedCount += 1
        if ACARSEncryptedCount > 150:
            break

    #Encrypt all CPDLC objects
    for count in range(0,len(self.CPDLCMessageList)):
        self.EncryptedObjectArray.append(self.CryptoHandler.EncryptMessageObject(self.CPDLCMessageList[count]))
        CPDLCEncryptedCount += 1
        if CPDLCEncryptedCount > 150:
            break
```

Figure 28 - Screenshot of function mid-execution, showing encryption process

Figure 28 shows the function used to encrypt messages, paused at the end of it's execution. As can be seen in the grey text at the top of the screenshot, 300 test messages, 150 ACARS and 150 CPDLC, have been encrypted and instantiated into objects. Once instantiated, these objects are appended to an EncryptedMessage array.



```
self.EncryptedObjectArray

                                                              Use Ctrl+Shift+Enter to add to Watches
Result:
v result = {list: 302} [(<CryptoHandler_2.EncryptedMessage object at 0x000001713C6B4130>, b'9QJ8fL7OdfYDI1Clk9ptelUtlxHzK4w... View
  v 000 = {tuple: 2} (<CryptoHandler_2.EncryptedMessage object at 0x000001713C6B4130>, b'9QJ8fL7OdfYDI1Clk9ptelUtlxHzK4woaX_I
    v 0 = {EncryptedMessage} <CryptoHandler_2.EncryptedMessage object at 0x000001713C6B4130>
      v EncryptedMessage = {bytes: 1036} b'gAAAAABiIi08F3QLH5aiVXbO_hrCw5EevXBRwGPfBsrttvW19RIrlufWHjNtJRP1MFr... View
          ICAOAircraftIDHash = {int} 2854713959041555310
        > Key = {bytes: 44} b'9QJ8fL7OdfYDI1Clk9ptelUtlxHzK4woaX_IG83_DZQ='
          MessageType = {str} 'ACARS'
        > RawMessage = {dict: 1} {'vdl2': {'app': {'name': 'dumpvdl2', 'ver': '2.1.1'}, 't': {'sec': 1645434277, 'usec': 551328}, 'freq': 1... View
          Source = {str} 'LOCAL'
```

Figure 29 - Evaluation of EncryptedObjectArray

Figure 29 shows the EncryptedObjectArray in the built-in debugger, halfway through execution. This shows the first element of the array consists of a CryptoHandler_2.EncryptedMessage object, and the associated encrypted message stored within the object. This highlights the correct operation of the algorithm.

5.6 - Message Transmission

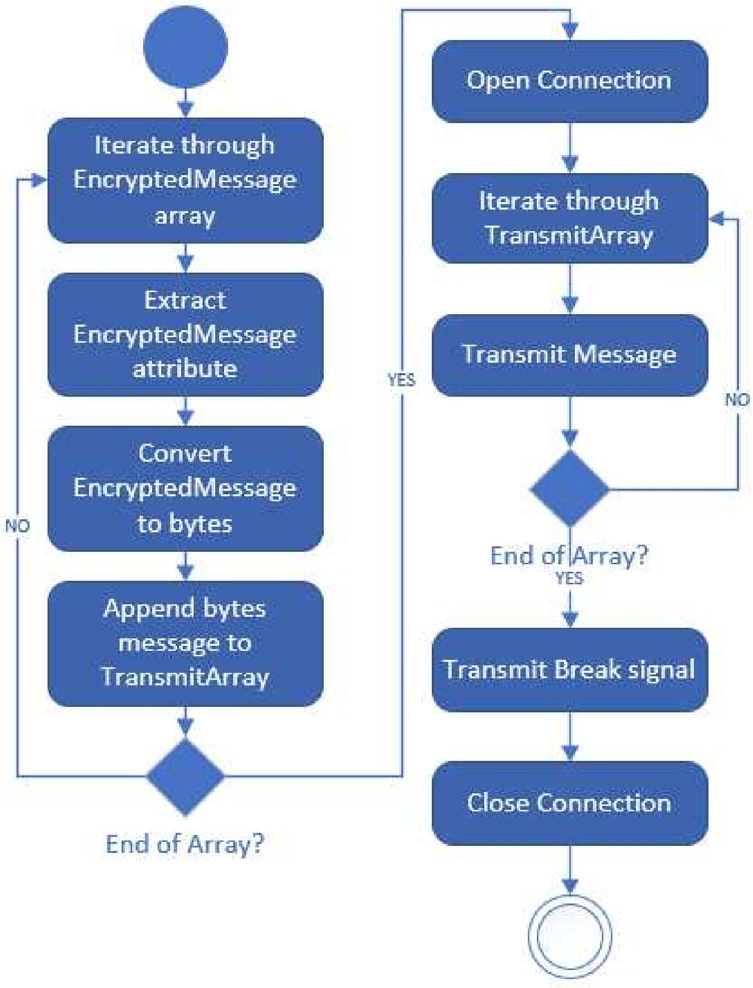

Iterate through EncryptedMessage array
Extract EncryptedMessage attribute
Convert EncryptedMessage to bytes
Append bytes message to TransmitArray
End of Array?
NO
YES
Open Connection
Iterate through TransmitArray
Transmit Message
End of Array?
NO
YES
Transmit Break signal
Close Connection

Figure 30 - Activity Diagram for Message Transmission

Figure 30 shows the process in which the artefact transmits encrypted messages to a remote instance.

As part of this message transmission process, the encrypted message keys, along with the identifying ICAOAircraft hash, are transmitted "out-of-band". This is done by writing the keys and hashes to a comma-separated values file. This is then read separately by the receiver to decrypt the messages that are transmitted.

The first step in transmitting the encrypted messages is to iterate through the array of EncryptedMessage objects. For each object, the raw encrypted message attribute 'RawEncryptedMessage' is extracted and converted from the string format to the bytes format. This bytes-string is then appended to the transmit array 'TransmitArray'. This process is repeated for every EncryptedMessage object in the EncryptedMessage array.

Once each message has been converted, a connection to the receiver is opened using the NetHandler file functions. The TransmitArray is then iterated through, with each bytes formatted message sent across to the receiver. Once each message has been sent, a bytes-encoded string: 'BREAK' is sent, signifying the end of message transmission. The sender then closes the connection and exits.

5.7 - Message Reception

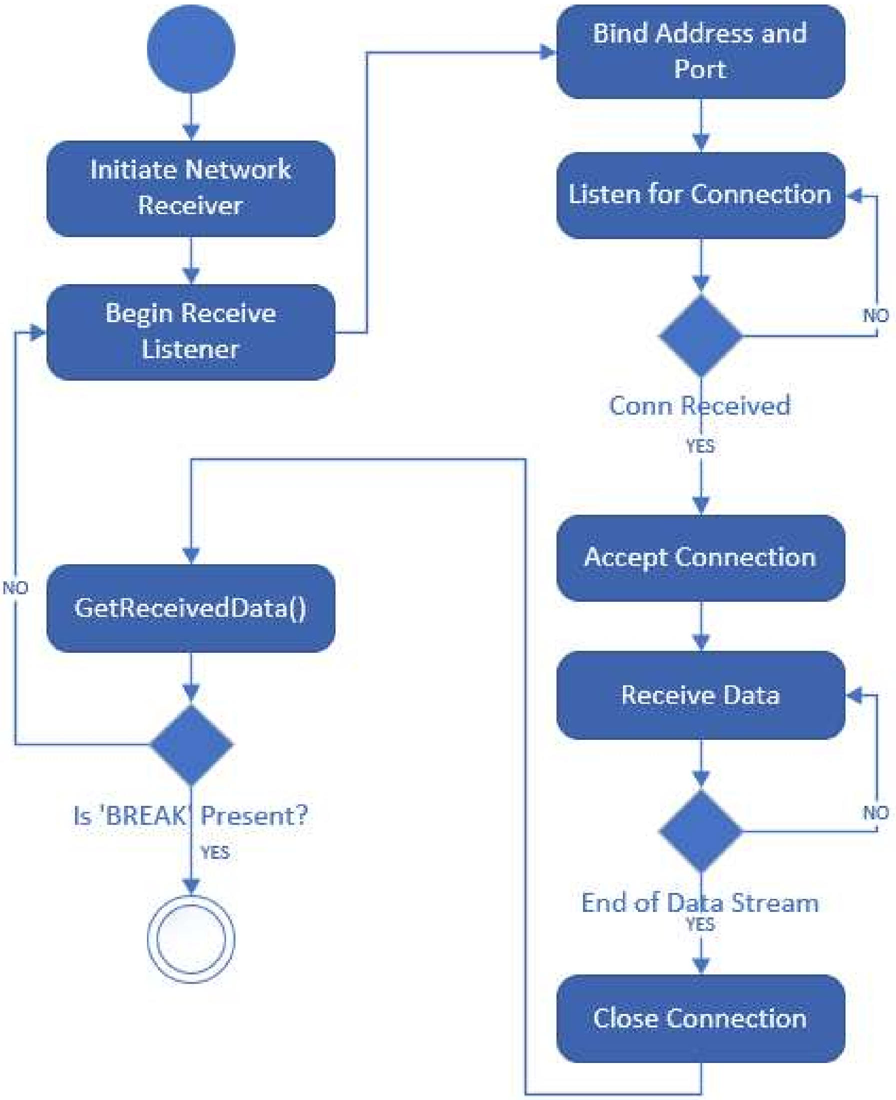


Figure 31 - Activity Diagram for message reception process

Figure 31 shows the process in which messages are received from a remote instance of the solution. First, a network receiver, implemented in the "NetHandler" file, is instantiated. This deals with the socket creation and listener, allowing the main program to handle data.

Once the socket is instantiated, and the address and port to listen are bound, the receiver waits for a connection from the transmitter. Once this connection is received, the data sent from the transmitter is stored in a temporary array - 'RawDataBuffer'. When the transmitter is finished, it will transmit a bytes-encoded string of 'BREAK', signifying to the receiver that the last message has been transmitted.

Once the data stream has been closed, the main program will run the GetReceivedData() function, which transfers the data stored in the RawDataBuffer into the main program arrays. The data, consisting of raw encrypted messages, is then used along with the keyfile, written by the transmitter, to decrypt and re-sort the messages.

5.8 - Message Decryption

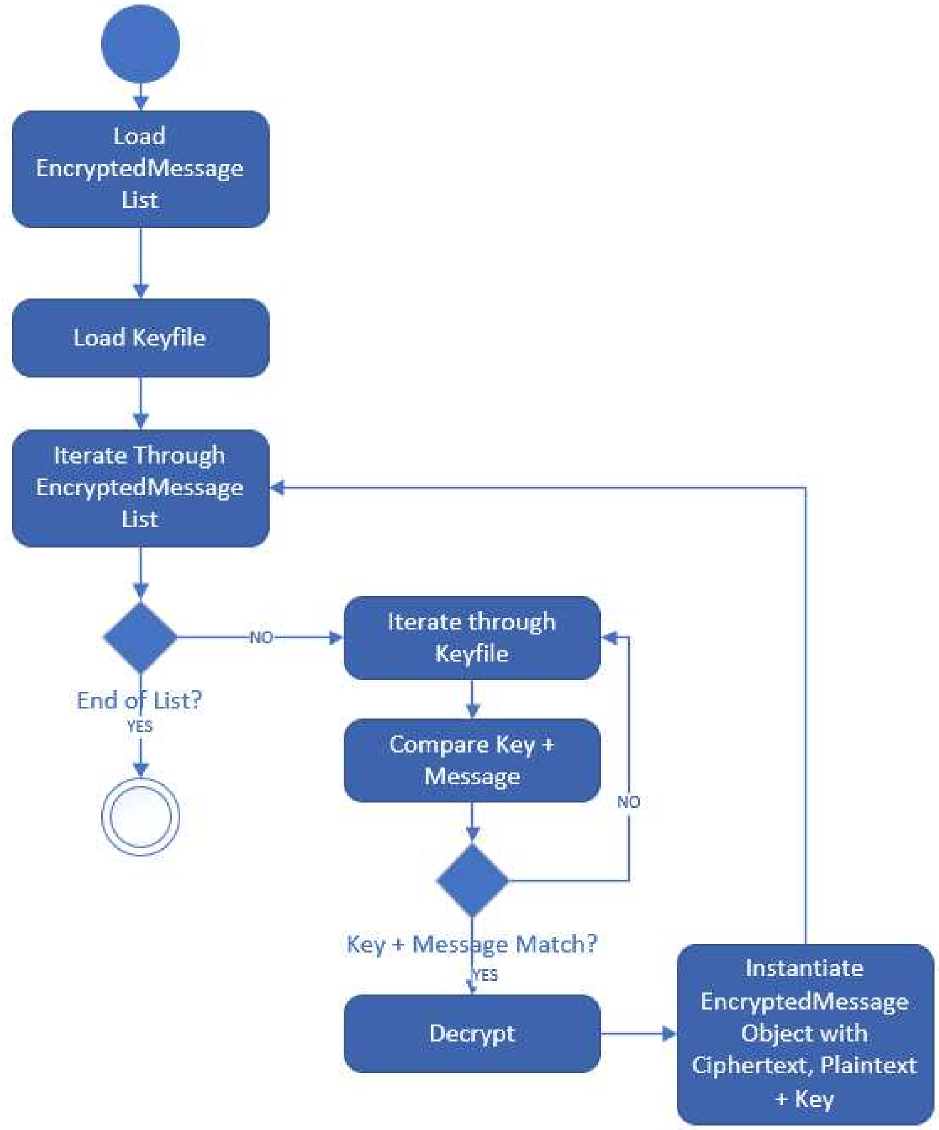
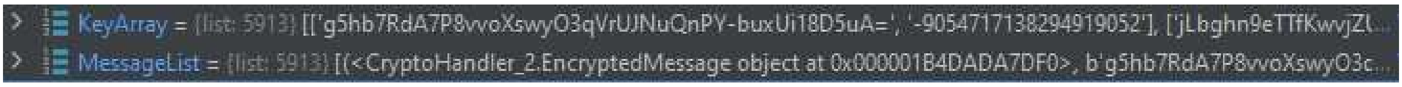

Load EncryptedMessage List
Load Keyfile
Iterate Through EncryptedMessage List
NO
End of List?
YES
Iterate through Keyfile
Compare Key + Message
NO
Key + Message Match?
YES
Decrypt
Instantiate EncryptedMessage Object with Ciphertext, Plaintext + Key

Figure 32 - Activity Diagram of Decryption Process

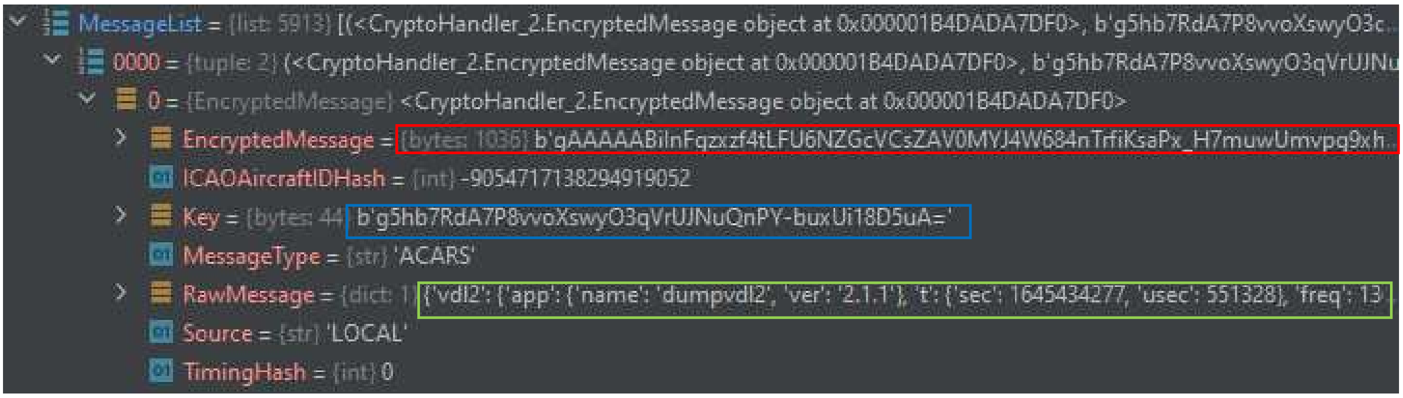
The message decryption process utilises a two-dimensional 'for' statement, where for each message within the EncryptedMessage list, every key in the keyfile will be tried against that message in an attempt to decode the message. If the key fails, the next key is tried, until the decryption of the message is successful. The algorithm then moves to the next message within the EncryptedMessage list, and the process is repeated. Whilst this is a logically inefficient approach, the size of the keyfile and the messages make the processing load manageable. Additionally, if this system were implemented in an actual aircraft, there would only be a single key associated with that aircraft, making this process trivial. More significant processing would be required on the ground network, where many encrypted messages may be received at a time, however the ground network is not limited by processing requirements in the same manner as an aircraft.



```
> KeyArray = {list: 5913} [['g5hb7RdA7P8vvoXswyO3qVrUJNuQnPY-buxUi18D5uA=', '-9054717138294919052'], ['jLbghn9eTTfKwvjZL...
> MessageList = {list: 5913} [(<CryptoHandler_2.EncryptedMessage object at 0x000001B4DADA7DF0>, b'g5hb7RdA7P8vvoXswyO3c...
```

Figure 33 - Snapshot of KeyArray and MessageList.

Figure 33 shows a snapshot of the KeyArray and MessageList - this clearly shows the list of EncryptedMessage objects and Keys that are used during the decryption process



```
v MessageList = {list: 5913} [(<CryptoHandler_2.EncryptedMessage object at 0x000001B4DADA7DF0>, b'g5hb7RdA7P8vvoXswyO3c...
  v 0000 = {tuple: 2} (<CryptoHandler_2.EncryptedMessage object at 0x000001B4DADA7DF0>, b'g5hb7RdA7P8vvoXswyO3qVrUJNu
    v 0 = {EncryptedMessage} <CryptoHandler_2.EncryptedMessage object at 0x000001B4DADA7DF0>
      > EncryptedMessage = {bytes: 1036} b'gAAAAABilnFqzxzf4tLFU6NZGcVCsZAV0MYJ4W684nTrfiKsaPx_H7muwUmvpq9xh...
        ICAOAircraftIDHash = {int} -9054717138294919052
      > Key = {bytes: 44} b'g5hb7RdA7P8vvoXswyO3qVrUJNuQnPY-buxUi18D5uA='
        MessageType = {str} 'ACARS'
      > RawMessage = {dict: 1} {'vdl2': {'app': {'name': 'dumpvdl2', 'ver': '2.1.1'}, 't': {'sec': 1645434277, 'usec': 551328}, 'freq': 13...
        Source = {str} 'LOCAL'
        TimingHash = {int} 0
```

Figure 34 - Detail on an EncryptedMessage object

Figure 34 shows the attributes of an EncryptedMessage object, showing the encrypted message string, associated key, and decrypted message.

5.9 - Testing Comparison

The purpose of this subsection is to utilise the objective list defined at the beginning of the project as a means to compare and contrast the solution, in an attempt to measure the overall effectiveness of the solution at addressing the identified problem.

#	Objective	Weight	Met
0	Receive and capture datalink messages	MUST	Yes - Strongly
1	Decode datalink messages	MUST	Yes - Strongly
2	Output readable files for further processing	MUST	Yes - Strongly
3	Output files in a well known format	SHOULD	Yes
4	Sort and categorise messages for further analysis	COULD	No
5	Allow the viewing, management and analysis of messages	COULD	No
6	Use a suitable cipher to encode the messages	MUST	Yes - Fernet

7	Use a suitable key management method to transfer the keys between entities	SHOULD	Partially - Requires “separate entity”
8	Use a suitable transmission medium to transfer messages	MUST	Partially - Network
9	Messages should be able to be transferred bidirectionally	COULD	No
10	Messages should be received correctly on the remote instance	MUST	Yes
11	Messages should be decrypted using the correct key	MUST	Yes
12	Intercepted messages should not be able to be decoded without the key	MUST	Yes

6. Encryption Performance

6.1 - Overview

A key aspect of the artefact is the performance of the encryption layer. Heavy cryptographic operations may not be possible on the processors that are present even in modern aircraft. Due to differing design requirements, a focus on reliability over new features, and the extensive certification process involved in creating new hardware and software, many aircraft use comparatively outdated hardware as part of their computing infrastructure.

For example, the Collins CMU-900, a modern communications management unit (CMU), found on new aircraft such as the Boeing 787, uses an Intel 486 processor for airline-specific applications, such as datalink message processing. (Koschar, 2020)

Considering this, the purpose of this section is to determine the nature of the cryptographic operations that are used as part of the Fernet encryption method, and the potential impact on running these operations on older hardware.

6.2 - Performance Testing and Profiling

6.2.1 - Performance testing methodology

Python is equipped with profiling tools, designed to analyse the activity of python scripts from a performance standpoint. Function calls, CPU time and other metrics can be gained through the use of the 'CProfile' module. For the purposes of performance analysis in the artefact, this module is used.

In addition to this, an external monitoring program can be used to extract performance information and related metrics. This is better than using a built-in monitoring framework, as the overhead of the framework can also be taken into account.

#	Test	CPU Usage	Memory Usage	Function Calls	Exec Time
0	Single-message encryption	0.765s	76k	38610	0.106
1	All Message encryption (6050)	2.25s	91mb	1157675	1.66s
2	Single-message decryption	0.734s	71k	38805	0.107

3	All message decryption (6050)	1:07:33.781	86mb	2937596151	3725s

Table 7 - Performance of cryptographic operations relating to the encryption layer

Table 7 shows the performance data collected, focusing on the cryptographic operations carried out within the solution. As can be seen from the above table, the most intensive operation is the decryption of all messages. This is partly due to the two-dimensional for loop utilised as part of the algorithm, requiring a high number of function calls to check each key against every message. This can be considered a ‘worse case’ scenario, as in a real-world scenario only one key will be held by an aircraft, negating the need to iterate through every key on the system to perform decoding.

6.3 - Encryption Strength

According to the Fernet specifications, the parameters used for cryptographic operations are as follows:

Key Used

The encryption key used, also known as a *Fernet Key*, is a Base64 encoding of two fields - Signing-key, and Encryption-Key (tmaher, 2014). Both of these fields are 128-bits long.

Whilst stronger implementations of AES exist, namely AES-256, additional complexity is often only use to increase the ‘margin of error’ - or resistance to attacks that utilise other information sources to narrow down the potential key pool, and therefore the amount of time it takes to brute-force the key. Due to the nature of the environment that this algorithm is designed to operate in, this additional level of complexity is likely to bring little added benefit when the additional cryptographic overheads are taken into consideration.

7. Critical Evaluation

7.1 – Overview

This section is primarily focused on discussing the overall success of the project, in academic terms rather than to the degree in which the solution addresses the problem domain. A critical review of the overall success will be given, as well as potential improvements to the overall project process, should it be repeated. In addition to this, the future research directions and works will be discussed, highlighting directions for further work and features should the project be extended. Finally, the overall value of the project in terms of academic skill and experience is reviewed.

7.2 – Project Success

In regards to the academic performance of the project, a number of factors need to be considered. From a theoretical and research standpoint, the project makes a good attempt at satisfying the objectives listed in the earlier sections. Despite issues relating to the availability of some technical materials, notably the ARINC standards, the existing literature available makes it possible to give an insight and overview of the field, highlighting a potential gap in the industry (security). In this regard, the project has had a degree of success. This is backed up by the confirmation of the security flaw suggested in the research, through practical analysis of messages and extraction of the data within. To this end, the project achieved more than initially expected, with the shear quantity of messages captured beyond what was initially expected, based off the capture hardware used.

However, despite these areas of success, there were several shortcomings and compromises that had to be made within the project. For example, the value of the artefact itself is limited in the sense that due to the unavailability of ARINC standards, the exact formatting of data cannot match that used on actual aircraft. Additionally, whilst the artefact deals satisfactorily with the processing of ACARS and CPDLC data, there is a significant body of other message types and formats, as well as additional systems that were not considered. The scope of these datalink systems and the governing bodies involved make it difficult to research effectively, and this is a potential area where the project struggles.

Related to this, in some cases non-academic sources were the only source of any information on certain topics. In this regard, these sections have questions raised regarding the original source of the information used.

Another potential metric was the effectiveness of the project management strategy in carrying out the project. Overall, the plan itself, whilst initially a valuable asset in managing time and getting an idea of the 'bigger picture' in relation to the project, became harder to utilise as time went on. By the end of the project, due to a number of reasons, the project plan was found to be too rigid and intolerant of change. This is a nature of the project management strategy selected - a waterfall implementation, however at the start of the project when this was selected, it was assumed that due to the unchanging nature of the project requirements, a waterfall strategy would fit well. This however, failed to take into account the *research* nature of the project, which had a significant impact on deadlines and progress.

7.3 – Improvements – "What would be done differently"

If the project were to be completed again, based off the performance and experience of this iteration, a number of modifications could be implemented, in order to streamline and generally improve the overall experience.

Firstly, the approach to research would be more structured. Throughout the course of the project, due to the expansive nature of the problem domain, often areas would become 'sinkholes' where research could be done for hours, however the actual research content would be unrelated to the problem domain itself. For example, whilst attempting to research how the different datalinks worked, a number of hours were spent investigating and attempting to determine how the Aeronautical Telecommunication Network (ATN) was implemented, despite this having little more than a passing relevance to the problem domain. This is where a more structured approach to research topics would have been useful.

In addition to this, the project management approach, selected for it's suitability to rigid and unchanging requirements, was too restrictive. Whilst the requirements of the problem domain itself are relatively fixed, the project management approach failed to account for the scope of the project changing due to research developments. This instigated a considerable amount of *deadline creep*, where due to changing requirements in earlier sections, the timing in subsequent sections was considerably disrupted. This led to compromises having to be made in later sections, due to time constraints.

The previous two points also relate to another potential modification, namely a greater focus on strategy and management, rather than tackling research or practical implementation straight away. Spending more time on strategy and management would likely make the rest of the project easier to implement.

Finally, from a more technical standpoint, if the project was to be done again, due to the heavy reliance on object-orientated programming in the artefact, a language that may be more suited to this type of programming may perform better.

7.4 – Future Works

As part of the future developments for the project, several additional avenues may be explored. These relate to the overall compatibility with existing infrastructure, as well as testing and assessing the viability further through more accurate representation of the environment it is designed to operate in.

A key potential development for the project would relate to the compatibility with the wider infrastructure currently in place. For example, with access to the ARINC 429 standard, the solution could be developed to take in, manage, and output ARINC 429 standard data words. This would reduce the work needed to implement the solution on aircraft, as the compatibility with systems used on aircraft is increased.

Another potential avenue relating to compatibility testing would be to explore the use of a software based transmitter, such as the HackRF One (Great Scott Gadgets, n.d.), to carry out the process of encrypting, encoding and modulating a genuine radio message, where it can then be received and decoded through DumpVDL2.

Another key development that may be addressed given further time and investigation would be to specify more detail and design a key distribution system that would operate alongside the encryption layer. This would detail the process in which keys would be transferred to an aircraft pre-flight, and distributed amongst the ground station network for decoding.

Given further time and research, the solution could also be expanded to include other types of messages, such as the X.25 and X.225 messages that support the underlying ground station network, as well as ADS-C and ATN messages. As well as this, the decoding and data capture could be expanded to include messages sent across the Inmarsat and Iridium satellite constellations, using a suitable L-band satellite receiver. High-frequency datalink (HFDL) messages could also be captured and analysed, with the solution expanded and assessed in satellite and HF use cases.

7.5 – Value of Learning Process/Personal Benefits

Undertaking this project has allowed me to develop my understanding of how the many elements of a project fit together, and how issues or problems at the start of the project can cause issues further down the line. As well as this, I have gained an understanding of the importance of a project management strategy, and how issues in this area quickly cause issues in other elements and aspects of the project. In this case, I underestimated the phenomenon of 'deadline creep' - where changes to the requirements or scope of the project can cause knock-on delays throughout other sections. Looking back, the adage of "measure twice cut once" can be applied, I often found myself keen to get stuck in on the practical elements before the management or strategic base was properly laid. As a result, elements that were unrelated or outside the scope of the project had to be removed once implemented.

My understanding of critical academic investigation has also deepened, through investigating the source of research more thoroughly than before. Questioning the legitimacy of the journal rather than the paper itself has led me to disregard some otherwise promising research due to the nature of the publisher. In addition to this, limitations regarding access of technical standards, especially those written by ARINC, has led me to develop my searching skills, using advanced search operators to narrow down the types and sources of information.

As well as this, I feel that my understanding of this particular field - aeronautical datalinks - has deepened significantly. As well as this, cryptography and cryptographic operations were an area that I lacked knowledge in before starting the project, often finding the arithmetic or numerical operations challenging. I have had to face these as part of the project, and feel that my skills in these areas has grown in conjunction with the knowledge related to the project. This makes my knowledge of the overall subject of computer science more versatile, as these unrelated elements play key roles in many different areas.

8. References

A. Kaviyarasu, n.d. *ARINC 429 - Data Bus for Civil Aircraft,* Madras: Anna University.

acarsd, 2022. *ACARS - Free ACARS Decoder for Linux and Windows.* [Online]
Available at: https://web.archive.org/web/20211205092113/https://www.acarsd.org/
[Accessed 09 05 2022].

Adobe, n.d. *Waterfall Methodology - A Complete Guide | Adobe Workfront.* [Online]
Available at: https://www.workfront.com/en-gb/project-management/methodologies/waterfall
[Accessed 23 05 2022].

AEEC, 2018. *Aviation Cyber Security Efforts,* s.l.: s.n.

Airbus, 1999. *A330 Flight Deck and Systems Briefing for Pilots,* s.l.: Airbus.

Aleisa, N., 2015. A Comparison of the 3DES and AES Encryption Standards. *International Journal of Security and its Applications,* 9(7), pp. 241-246.

Anon., n.d. Manual on the Implementation of HFDL. In: s.l.:s.n., p. 5.

ARINC, 2002. Implementing the Critical Links. *The Global Link*, October, pp. 2-4.

ARINC, 2014. *618-7.* [Online]
Available at:
https://archive.ph/20140328055821/https://www.arinc.com/cf/store/catalog_detail.cfm?item_id=2072
[Accessed 05 05 2022].

ARINC, n.d. *Arinc Communications Addressing & Reporting System,* s.l.: ARINC.

AstonJet, n.d. *AstonJet.* [Online]
Available at: https://www.astonjet.com/en/
[Accessed 26 05 2022].

Atlassian, n.d. *What is Agile? | Atlassian.* [Online]
Available at:
https://www.atlassian.com/agile#:~:text=Agile%20is%20an%20iterative%20approach,small%2C%20but%20consumable%2C%20increments.
[Accessed 17 05 2022].

British Aerospace, 1997. *Improvements to VHF air-to-ground communication,* s.l.: IET.

Corentin Bresteau, S. G. P. B. J. M. F., 2018. On the Security of Aeronautical Datalink Communications: Problems and Solutions. *IEEE Xplore,* pp. 1A4-1 - 1A4-13.

Corentin Bresteau, S. G. P. B. J. M. F., 2018. *On The Security of Aeronautical Datalink Communications: Problems and Solutions,* Montreal: s.n.

El Gaabouri Ismail, C. A. R. N., 2020. *Fernet Symmetric Encryption Method to Gather MQTT E2E Secure Communications for IoT Devices,* Tetouan: ResearchGate.

Euronews, 2021. *HMS Defender: AIS Spoofing is opening up a new front in the war on reality.* [Online]
Available at: https://www.euronews.com/next/2021/06/28/hms-defender-ais-spoofing-is-opening-up-a-new-front-in-the-war-on-reality
[Accessed 15 01 2022].

FlightRadar24, 2022. *F-HATV - Cessna 680A Citation Latitude - AstonJet - FlightRadar24.* [Online]
Available at: https://www.flightradar24.com/data/aircraft/f-hatv
[Accessed 26 05 2022].

Free Management Books, n.d. *The Six Step Problem Solving Model.* [Online]
Available at: http://www.free-management-ebooks.com/news/six-step-problem-solving-model/
[Accessed 15 05 2022].

Great Scott Gadgets, n.d. *HackRF One - Great Scott Gadgets.* [Online]
Available at: https://greatscottgadgets.com/hackrf/one/
[Accessed 30 05 2022].

Haindl, B., 2007. *An Independent Technology Assessment for a Future Aeronautical Communications System based on Potential Systems like B-VHF,* Vienna: IEEE.

ICAO Inter-Regional SATCOM Voice Task Force, 2012. *Satellite Voice Guidance Manual (SVGM),* s.l.: ICAO .

ICAO, 2013. Controller-Pilot Data Link Communications (CPDLC). In: *ICAO Global Operational Datalink Document.* s.l.:ICAO, pp. 2-39, 2-40.

ICAO, 2015. *Manual on VHF Digital Link (VDL) Mode 2.* 2nd ed. Montréal: INTERNATIONAL CIVIL AVIATION ORGANIZATION.

ICAO, 2016. *Global Operational Datalink Manual.* 1st ed. Quebec: ICAO.

ICAO, 2016. ICAO Global Operational Datalink Manual. In: s.l.:s.n., pp. 1-7.

ICAO, 2020. *ICAO Cybersecurity Action Plan,* s.l.: ICAO.

ICAO, 2022. *eANP Vol II, Part III, Supplements - All Documents.* [Online]
Available at:
https://www.icao.int/eurnat/eur%20and%20nat%20documents/forms/allitems.aspx?RootFolder=%2FEURNAT%2FEUR%20and%20NAT%20Documents%2FEUR%20Documents%2FDoc%207754%20%2D%20EUR%20Region%20%2D%20eANP%20Vol%20II%2FeANP%20Vol%20II%2C%20part%20III%2C%20Supplements&F
[Accessed 10 05 2022].

IETF, 2007. *Internet Security Glossary, Version 2,* s.l.: IETF.

Institute of Organisational Development, n.d. *Six Steps to Effective Problem Solving Within Organisations.* [Online]
Available at:
https://webcasts.td.org/uploads/assets/3730/document/SixStepstoProblemSolvinginOrganizations.pdf#:~:text=In%20this%20article%2C%20we%20will,and%20create%20an%20action%20plan.
[Accessed 15 05 2022].

International Maritime Organisation, 2018. *SOLAS 2018 - Consolidated Edition,* s.l.: s.n.

ISO, 2010. *ISO/IEC 18033-3:2010 - Information Technology.* [Online]
Available at:
https://web.archive.org/web/20131203003348/http://www.iso.org/iso/home/store/catalogue_ics/catalogue_detail_ics.htm?csnumber=54531
[Accessed 06 05 2022].

ITU, 2000. *Nomenclature of the Frequency and Wavelength Bands Used in Telecommunications,* s.l.: ITU.

James J. Rooney, L. N. V. H., 2004. Root Cause Analysis for Beginners. *Quality Progress*, July, pp. 45-53.

James V. Harmon, L. D. M. F. (. a. L. J. R. L., 1984. Automated HF Communications. *Army Communicator*, pp. 22-26.

Jing Wang, Y. Z. J. D., 2020. ADS-B Spoofing attack detection method based on LSTM. *EURASIP Journal on Wireless Communication and Networking.*

Joshua Smailes, D. M. M. S. M. S. V. L. I. M., 2021. *You talkin' to me? Exploring Practical Attacks on Controller Pilot Data Link Communications,* Oxford, Zurich: ACM.

Kai Petersen, C. W. D. B., 2009. The Waterfall Model in Large-Scale Development. *International Conference on Product-Focused Software Process Improvement,* pp. 386-400.

Kaspersky, n.d. *What is Data Encryption?.* [Online]
Available at: https://www.kaspersky.co.uk/resource-center/definitions/encryption
[Accessed 06 05 2022].

Koschar, K., 2020. *What I Learned Trying to Hack a 737,* s.l.: DEF CON.

Lemiech, T., 2021. *GitHub - szpajder/dumpvdl2: VDL Mode 2 Message Decoder and Protocol Analyzer.* [Online]
Available at: https://github.com/szpajder/dumpvdl2
[Accessed 31 03 2022].

Lemiech, T., 2021. *szpajder/DumpVDL2: VDL Mode 2 Message Decoder and Protocol Analyser | Github.com,* s.l.: s.n.

Lundstrom, S., 2016. *Technical Details of VDL mode 2,* s.l.: s.n.

Martin Strohmeier, M. S. R. P. V. L. I. M., 2016. *On Perception and Reality in Wireless Air Traffic Communications Security,* s.l.: s.n.

Marzanah A.Jabar, F. S. M. H. S. A. A. A. G. &. H. I., 2009. An Investigation into Methods and Concepts of Qualitative Research in Information System Research. *Computer and Information Science,* 2(4), pp. 47-54.

Matthew Smith, D. M. M. S. V. L. I. M., 2017. *Economy Class Crypto: Exploring Weak Cipher Usage in Avionic Communications via ACARS,* s.l.: s.n.

Matthew Smith, D. M. M. S. V. L. I. M., 2018. *Undermining Privacy in the Aircraft Communications Addressing and Reporting System (ACARS),* s.l.: s.n.

Nana Yaw Asabere, W. K. T. S. H. G. A., 2014. Improving Computer Science Research in Polytechnic Education. *International Journal of Computer Science and Telecommunications,* 5(9), pp. 8-16.

Prince, B., 2012. *Air Traffic Control Systems Vulnerabilities Could Make for Unfriendly Skies.* [Online]
Available at: https://www.securityweek.com/air-traffic-control-systems-vulnerabilities-could-make-unfriendly-skies-black-hat
[Accessed 31 05 2022].

Robert W. Stewart, L. C. D. A. K. B. D. C. I. C. M. M. E. S., 2015. A Low-Cost Desktop Software Defined Radio Design Environment Using MATLAB, Simulink, and the RTL-SDR. *IEEE Communications Magazine*, September, p. 64.

Roy, A., 2000. *Security Strategy for US Air Force to use Commercial Data Link,* s.l.: s.n.

Signal Identification Wiki, 2016. *File:VDL2 Waterfallthmb.jpg.* [Online]
Available at: https://www.sigidwiki.com/wiki/File:VDL2_Waterfallthmb.jpg
[Accessed 31 03 2022].

Skybrary, n.d. *Aircraft Communications, Addressing and Reporting System.* [Online]
Available at: https://skybrary.aero/articles/aircraft-communications-addressing-and-reporting-system
[Accessed 25 05 2022].

Skyvector, n.d. *Skyvector: Flight Planning / Aeronautical Charts.* [Online]
Available at: https://skyvector.com/
[Accessed 06 05 2022].

Sun, S., 2009. *ACARS Data Identification and Application in Aircraft Maintenance,* s.l.: s.n.

The Boeing 737 Technical Site, n.d. *Communications.* [Online]
Available at: http://www.b737.org.uk/communications.htm
[Accessed 05 05 2022].

tmaher, 2014. *spec/Spec.md at master - fernet/spec - GitHub.* [Online]
Available at: https://github.com/fernet/spec/blob/master/Spec.md
[Accessed 27 05 2022].

UAS International Trip Support, 2016. *Datalink Communications - The Origins and Course of Air-to-Ground Messaging,* s.l.: s.n.

WAVECOM, 2022. *VDL-M2 Aeronautical Data Link - Advanced Protocols,* Zurich: Wavecom Elektronik AG.

Williams, L., 2007. *A Survey of Agile Development Methodologies,* s.l.: s.n.

Yoohwan Kim, J.-Y. J. S. L., 2017. ADS-B Vulnerabilities and a Security Solution with a Timestamp. *IEEE A&E Systems Magazine*, Nov, pp. 52-61.