MULTI-LAYER SECURITY MECHANISM FOR COMMERCIAL AIRCRAFT SOFTWARE DISTRIBUTION SYSTEM: CASE OF AIRLINES

A Thesis Presented

by

Getero Gaga Dabulo

to

The Faculty of Informatics

of

St. Mary's University

In Partial Fulfillment of the Requirements for the Degree of Master of Science

in

Computer Science

July, 2021

ACCEPTANCE

Multi-layer Security Mechanism for Commercial Aircraft Software Distribution System: Case of Airlines

By

Getero Gaga Dabulo

Accepted by the Faculty of Informatics, St. Mary's University, in partial fulfillment of the requirements for the degree of Master of Science in Computer Science

Thesis Examination Committee:

Asrat Mulatu (PhD)	[illegible]	**29/07/2021**
Internal Examiner	**Signature**	**Date**
Temtim Assefa	[illegible]	**28/07/2021**
External Examiner	**Signature**	**Date**
Getahun Semeon (PhD)	[illegible]	**28/07/2021**
Dean, Faculty of Informatics	**Signature**	**Date**

July 09, 2021

DECLARATION

I, the undersigned, declare that this thesis work is my original work, has not been presented for a degree in this or any other universities, and all sources of materials used for the thesis work have been duly acknowledged.

Getero Gaga Dabulo

Full Name of Student

[illegible]

Signature

Addis Ababa
Ethiopia

This thesis has been submitted for examination with my approval as advisor.

Getahun Semeon (PhD)

Full Name of Advisor

[illegible]

Signature

Addis Ababa
Ethiopia

July 09, 2021

Acknowledgments

First of all, I praise God for giving an opportunity and helping me to complete this thesis. Next to God, I would like to thank and express my heartfelt appreciation to my advisor, Dr. Getahun Semeon. His support, advise, and guidance has been invaluable during the research and writing of this thesis. The successful completion of this thesis would not have been possible without his help. I would also thank my friends and colleagues for their expertise and sharing ideas. Finally, my special thanks go to my family, especially to my wife for her love, positivity, and encouragement.

Table of Contents

List of Acronyms/Abbreviations

AADS	Airplane Assets Distribution System
ARINC	Aeronautical Radio INC.
CISA	Cybersecurity and Infrastructure Security Agency of USA
COTS	Commercial-Off-The-Shelf
CSET	Cyber Security Evaluation Tool
EASA	European Aviation Safety Agency
FAA	Federal Aviation Administration
GPS	Global Positioning System
IATA	International Air Transport Association
ICAO	International Civil Aviation Organization
IDPS	Intrusion Detection & Prevention System
IT	Information Technology
LRU	Line Replaceable Unit
LSAP	Loadable Software Airplane Parts
MFA	Multi-Factor Authentication
MRO	Maintenance, Repair and Overhaul
NIST	National Institute of Standards and Technology
O-ESA	Open Enterprise Security Architecture
OSA	Open Security Architecture
RTCA	Radio Technical Commission for Aeronautics
STPA	Systems Theoretic Process Analysis
VPN	Virtual Private Network

List of Figures

List of Tables

Abstract

Aircraft avionics systems are one of the most critical components of an airplane due to their criticality for safe flight operations. These systems rely on loadable software aircraft parts to perform functions previously handled manually or by analog systems. When a new or update to an existing software is needed to be installed on the aircraft, the software parts are transferred from the manufacturer/supplier to operators. Operators should have a software management process that includes adequate protections from software tampering while the software is in storage and during transfers.

Currently, there is no security mechanism that can prevent installation of software from non-official sources for the previous design airplanes that did not adapt technology advancements. The attackers could take advantage of this vulnerability and tamper the software parts, which could negatively affect the safe operation of the airplane.

To solve this problem, this study employed the design science research methodology, which is a rigorous research framework that creates and evaluates information technology (IT) artifacts, to solve the identified security problems. This study performed a comprehensive security analysis of the aircraft software distribution systems by applying the systems approach called Systems Theoretic Process Analysis for Security.

This study found out two critical security vulnerabilities in the aircraft software distribution system: (1) there is no security mechanism for the previous design airplanes to authenticate the identity of the sender of the software and to ensure that the original content of the document is unchanged and (2) password-based single-factor authentication is used for accessing the ground-based software servers as well as maintenance laptops.

Finally, this study demonstrated that the identified vulnerabilities could be eliminated or prevented from being exploited by applying the proposed solutions. Therefore, the major contribution of this study is applying a multi-layer security mechanism for the aircraft software distribution system, which enhances the existing security mechanisms and provides adequate security protection.

CHAPTER ONE - INTRODUCTION

1.1. Background of the Study

Aviation is a vital industry that contributes substantially to economic development and improved living conditions. According to the ICAO, the 4.1 billion passengers transported in 2017 are expected to grow to around 10 billion by 2040. And according to IATA, 35% of world trade by value is transported by air cargo, equivalent to $6.4 trillion of goods. The role of the aviation industry in commerce, trade and transport infrastructure makes it indispensable to the global economy [1].

According to the US Cybersecurity and Infrastructure Security Agency (CISA) Aviation is part of Critical National Infrastructures whose assets, systems, and networks, whether physical or virtual, are considered so vital that their incapacitation or destruction would have a debilitating effect on security, national economic security, national public health or safety, or any combination thereof [2].

According to IATA study on the importance of air transport and tourism to Ethiopia, air transport and foreign tourists arriving by air are the significant economic enablers that currently support 5.7% of the nation's GDP valued at USD 4.2 billion and about 1.1 million jobs. If current trends persist, Ethiopia's air transport market will expand by 226% over the next 20 years, with annual passenger journeys increasing from 7.2 million in 2017 to 23.5 million a year by 2037 [3].

One key characteristic of the aviation industry is the high level of inter-dependency between the various sectors of activity (airports, air navigation services, airlines, etc.), and interconnectivity with related systems (maintenance services, network connectivity services, fuel distribution systems, etc.). One incident at any point in this value chain can have severe consequences in other areas [1]. The aviation industry sits within a broader geopolitical landscape that includes intergovernmental policies, security concerns, national interests, economics, and society.

The never-ending improvement of new digital technology has given birth to a new generation of aircraft that implement a remarkable number of new technologies such as IP-enabled networks, COTS (commercial-off-the-shelf) components, wireless connectivity, and global positioning systems (GPSs) [7]. The use of information technology (IT) to integrate airline systems is called Electronic Enabling (e-Enabling). The move to e-Enabled aircraft is being driven by the need to achieve greater efficiency and flight volume, lower cost, and an improved passenger experience [4] [5]. Airlines throughout the world are expanding their use of information technology within their maintenance, engineering, and flight operations organizations. Aircraft manufacturers, such as Boeing and Airbus, offer several e-Enabled tools and services, as well as the expertise and guidance to help airlines implement and integrate e-Enabled systems.

The aviation ecosystem faces increasing risks to flight safety from a complex and diverse set of threats. In particular, the growing connectivity between airplane networks and systems and various other systems via the Internet increasingly presents more opportunities for cyberattacks [6]. For example, critical data used by cockpit systems could be altered, someone with authorized access could intentionally or unintentionally misuse flight data, commercial components within avionics systems could contain vulnerabilities that enable cyberattacks, and malevolent hackers could seek to disrupt flight operations with various types of attacks on navigational data.

The evolving cyber threat landscape, combined with the increasing use of internal networks on airplanes and the increasing connections between airplanes and external sources, could lead to increasing risks for future flight safety. Among others, cyber threats pose increasing risks to avionics systems. Cyber threats, which include any circumstances or events with the potential to have an adverse impact on cybersecurity, can be intentional or unintentional and can come from a variety of sources. Unintentional threats can come from anyone and anywhere, while intentional threats can include criminal groups, hackers, disgruntled employees, foreign nations engaged in espionage and information warfare, drug trafficking organizations, and terrorists.

Modern aircraft systems rely on software to perform functions previously handled manually or by analog systems [11]. Airplanes are increasingly reliant on complex software that may have security vulnerabilities potentially could be exploited by those with criminal intentions. Vulnerabilities could occur due to modifications (patches) to commercial software not being applied, insecure supply chains, malicious software uploads, outdated systems on legacy airplanes, and flight data spoofing attacks [6]. Airplane systems may be built with commercial off-the-shelf software and components, which may support a variety of functions on board the airplane, including the maintenance and crew devices that connect to them. If not completely isolated from external networks, such software will likely need to be updated on a continuous basis to respond to newly identified vulnerabilities and changing threat scenarios [6].

Aircraft software distribution system involves multiple entities with different roles, including avionics suppliers, airframe manufacturer, airline, and aircraft [9]. Federal Aviation Administration (FAA) is responsible for the safety and oversight of commercial aviation, and it recognizes avionics cybersecurity as a potential airworthiness and safety issue for e-enabled commercial transport airplanes [6]. It stresses the criticality of some of the software onboard airplanes through well-established guidance assuring their proper design and development for continued airworthiness [12], e.g., RTCA DO-178B/C. Yet the guidance does not even address the issue of software distribution and its security [10].

1.2. Statement of the Problem

The current aviation regulations and guidelines do not specifically address the issue of software distribution and its inherent security. Operators are responsible to have a software management process that includes adequate protections from software tampering while the software is in storage and during transfers [11]. Operators have to perform security assessment and, select and demonstrate security mechanisms that can provide adequate protection for their aircraft software management process (i.e., receiving, storing, transporting, and loading of software).

Currently, there is no security mechanism that can prevent installation of software from non-official sources for the previous design airplanes that do not have technology and capability for the digital signatures [28] [6]. The installed software might have been tampered with by an attacker before installation, without changing the software version number or falsifying it as an official version [6] [15]. Besides this vulnerability, single factor authentication that is username and password is applied in airlines for controlling access to the data distribution and software storage servers and maintenance laptops. Nowadays, a password-based single factor authentication (i.e., username and password only) is no longer considered secure because it can easily be compromised, and bad actors can take advantage of it to tamper the software [40] [42].

The review of previous studies showed that the security analysis of avionics systems software distribution was not comprehensive, and it did not cover all aircraft types, specifically the previous design airplanes that did not adapt the technology advancements [10,14,15]. Also, the research solutions of previous studies did not take the existing software distribution infrastructures and technologies into account [10,14,15].

Therefore, the purpose of this study is to identify the key security issues associated with the aircraft software distribution system, and to design an artifact, i.e., security architecture, by applying multi-layer security mechanisms that will enhance the existing security mechanisms to provide adequate protection.

1.3. Research Questions

The following research questions are formulated to address the research problem:

- What cybersecurity vulnerabilities exist in the current aircraft software distribution system in the commercial aviation industry, and specifically in airlines?
- What cybersecurity mechanisms can be applied to adequately protect the aircraft software distribution system in airlines?

1.4. Objectives

1.4.1. General Objective

The general objective of this research is to develop an efficient security architecture by applying a multi-layer security mechanism in order to adequately protect the airlines aircraft software distribution system from tampering.

1.4.2. Specific Objectives

- ➢ To analyze the cybersecurity mechanisms that are currently applied to the aircraft software distribution system in airlines for potential vulnerabilities.
- ➢ To design an artifact, i.e., security architecture, for aircraft software distribution system using the multi-layer security mechanism.
- ➢ To evaluate the proposed security mechanism and architecture to ensure the identified security issues have been properly addressed.

1.5. Methodology

A rigorous research methodology which is design science research is applied in carrying out this research. Design Science Research is a research framework that creates and evaluates information technology (IT) artifacts intended to solve identified organizational problems [27]. This is in line with the goal of this study, which is to develop a new security architecture for the aircraft software distribution system by applying multi-layer security mechanism. This study adopts Systems Theoretic Process Analysis for Security (STPA-Sec) to carry out a comprehensive security analysis of the airlines aircraft software distribution system in order to identify potential security vulnerabilities and their overall impact. The findings of the STPA-Sec analysis will be used to design the artifact.

1.6. Significance of the Study

The major contribution of this study is applying a multi-layer security mechanism for the aircraft software distribution system, which enhances the existing security mechanisms and provides adequate protection. This will improve the integrity of the software parts distribution process, protect the integrity of software parts throughout their lifecycle, as well as provide strong access controls and authentication mechanisms for ensuring integrity. Therefore, this study benefits the airline management as well as the security personnel.

1.7. Scope and Limitation of the Study

In line with the objective of the study, this thesis aimed at enhancing the cybersecurity mechanisms for aircraft systems' software distribution system. The scope of this study is limited to the security of the aircraft systems software distribution system, which includes software receiving, storing, transporting, and loading processes. The security of software development is outside of the scope of this study. Even though the enterprise architecture is a holistic roadmap of the enterprise that includes business processes and IT (information technology) infrastructures, this study does not cover the corporate level IT security and enterprise architecture. Rather it is limited to the security and security architecture of the LSAP distribution system. Due to the time constraint this study was based on general airlines, and it did not study a specific airline software management process. Hence, this could be considered as the limitation of this study.

1.8. Organization of the Study

This study consists of six chapters. Chapter one focuses on the background of the study, problem statement, objectives, significance, and scope and limitation of the study. In Chapter two, a range of literatures review is captured that gathers relevant information regarding the cybersecurity issues in the commercial aircraft systems' software distribution system. In chapter three, detail of methodology followed to achieve results is outlined. Chapter four presents comprehensive security analysis of the loadable software airplane parts (LSAP) distribution system. Chapter five contains the proposed multi-layer security mechanism and security architecture for LSAP distribution system. Chapter six evaluates the proposed conceptual security mechanism. Chapter seven focuses on main findings, conclusion, and recommendations of the study.

CHAPTER TWO -
LITERATURE REVIEW

2.1. Overview

This chapter provides an overview of major concepts and a review of previous studies on aircraft systems' software distribution system and its inherent security limitations. The first part of this chapter gives a description of the loadable software aircraft parts (LSAP) and types of loadable software. The second part presents a brief overview and architecture of aircraft software distribution system. The third part of this chapter deals with aircraft systems security. In the fourth part, current regulations and guidelines pertaining aircraft software security are discussed. Finally, selected previous works on Cybersecurity Mechanism for Aircraft Systems' Software Distribution System are reviewed and gaps identified and summarized.

2.2. Loadable Software Aircraft Parts (LSAP)

In aviation, the term Loadable Software Aircraft Parts (LSAPs), also referred to as Field Loadable Software (FLS), describe any piece of software that can be exchanged or updated without the need for replacing the hardware - Line Replaceable Unit (LRU). Software is data or code that defines controls or is used by aircraft systems to perform an intended function. Since first introduction to commercial aviation in the 1970s, LSAPs have become the de-facto standard [15]. Modern airplanes feature avionics systems whose functionality may be changed or updated using onboard loadable software. LSAPs are designed for transferring into target hardware without any physical alteration. Modifying system functionality with new software instead of with modified or new hardware helps operators reduce the total number of hardware LRUs in inventory, increase hardware commonality, and reduce airplane modification time.

Virtually all the software is airplane loadable, which allows for flexibility in adding new functionally, fixing software problems, or modifying software to accommodate re-designed hardware. But on the other hand, it also adds complexity to the airplane configuration management process. There are numerous reasons why a given software part could change during the lifetime of the airplane. The configuration of the airplane must be known and tracked during all of these changes to maintain operational approval for the airplane.

The individual component parts of aircraft systems often require specific software configurations and, in some cases, specific interface settings to allow an aircraft system to function correctly. It is important that the software installed in aircraft systems is maintained in a configuration relevant to that particular aircraft. Software eligibility can differ depending on the aircraft configuration or serial number. This can mean that different configurations of software are required for aircraft in a fleet that are of the same make and model.

LSAPs are considered part of the approved design. LSAPs require the same controls as physical aircraft parts in that they are serviceable, conform to design specifications and are eligible for fitment. Provisions should be established during any data retrieval processes to ensure that data corruption does not affect the software. A confirmed link must exist between the software and the applicable LRU. Electronic distribution of software requires protection against unauthorized security breaches.

The earliest forms of LSAPs used in commercial aviation were loaded with floppy disks. Over the years, floppy disks were replaced with portable media of higher capacity, such as compact discs (CDs) and portable hard drives. Currently, the state of the art is to send the software to airlines over the Internet. The airline in turn distributes it to the aircraft over wireless networks. Once the software is on the aircraft, mechanics can send it to the LRU whose software needs to be updated.

2.2.1. Types of Loadable Software

There are several types of software which are transferred to LRUs [41]:

- Operational Program Software (OPS)
- Operation Program Configuration (OPC) file
- Database
- Airline modifiable software

The Operational Program Software (OPS) is the set of program instructions for an LRU. The OPS is generally the largest (most words of memory) and most complex software in each LRU. As such, the OPS requires the most time to load and the most effort to certify a new version. Each version of an OPS has a unique software part number. An OPS part number configuration check is required following the installation of an LRU.

An Operation Program Configuration (OPC) file is the set of configuration data that determines the function of an LRU. An OPC is a special purpose database that enables or disables optional functions of features of the OPS. Hardwired discretes (program pins) are in use on many systems to define the desired customer unique options, equipment complements, airplane type, etc., to the OPS. The OPC is small in relation to the OPS, on the order of 100 bytes, thus the data transfer time is usually less than one minute.

A database is a collection of data easily arranged for access, retrieval, and update by an LRU's operating system. Some of the databases used by LRUs are [41]:

- FMC Navigation Database (NDB)
- FMC Model/Engine Database (MEDB)
- FMC Performance Defaults Database
- FMC QRH T/O Speeds Database
- ACARS Database (ACARS MU)
- CDS Display Unit Database (DEU)
- DFDAU Mandatory Database
- DFDAU ARINC 429 Broadcast Database

Airline modifiable software is a data file used by some LRUs to provide information used by the OPS. On some airplanes, when data is to be recorded or formatted, reports generated, seating zones services specified, etc., the appropriate OPS refers to the airline modifiable software for the necessary information to accomplish the task. The affected systems are designed to use airline modifiable software data files, which are generated by the airlines, to allow customization of these features. Airline modifiable software is typically characterized by being data rather than programs or executable code. The scope of modifiability of airline modifiable software is controlled by the LRU's certified OPS, which is required to prevent user modifications from affecting safety, whether or not they are correctly implemented. On this basis such modification is permitted without certification authority review.

2.3. Aircraft Systems Software Distribution System

Aircraft systems' software distribution system involves multiple entities with different roles, including avionics suppliers, software vendors, aircraft manufacturer, aircraft owner/ operator, maintenance provider, aircraft, and different equipment & tools. The electronic distribution of airplane information assets, i.e., software and data, can be modeled by a generic system, which is commonly called Airplane Assets Distribution System (AADS). Figure 1 illustrates the AADS model with the entities and flow of assets. Not all entities interact directly with all others. Note that functional overlaps are possible, with a single entity assuming roles of multiple entities, e.g., an airplane manufacturer can be a supplier for some software. The nature and content of interactions change depending on the lifecycle state of a specific airplane, which can be in development, assembly, testing, use, resale, etc.

The responsibility of the AADS for an asset begins when it takes over the asset from its producer, e.g., supplier or airplane manufacturer, and ends when it delivers the asset at its destination, i.e., embedded systems such as a Line Replaceable Unit (LRU) in an airplane, or at the consumer of airplane-generated data. The path between the producer and the destination of the asset is referred to herein as the end-to-end path. Each supplier is accountable to produce safety-assured software as per [12]. Each of the links in this path must fulfill the security objectives.

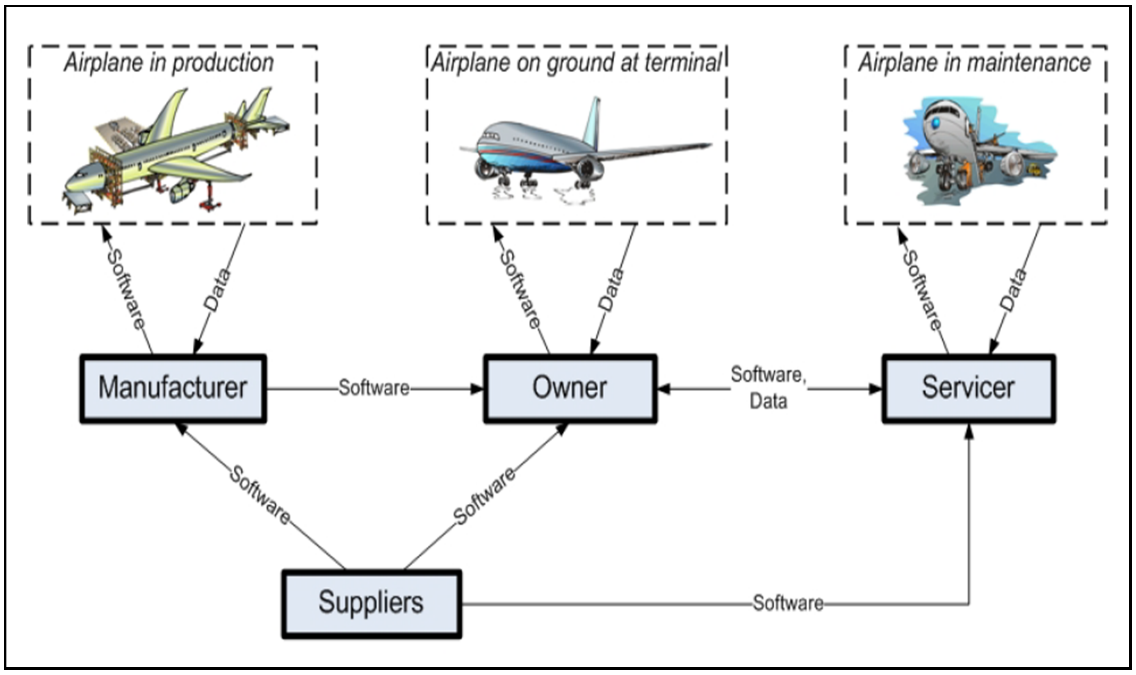

Figure 1. Airplane Assets Distribution System (AADS)
Source: [10]

2.4. Aircraft Systems Software Security

When avionics software was first introduced in the 1970s, security was only a concern insofar as the computers running the software had to be protected from physical sabotage. This was mainly due to two reasons: (1) most computers were not networked, and the Internet did not exist yet, and (2) avionics software and hardware used to be custom made for aviation. The systems were similar to more widely used commodity hardware and software in the PC mass-market, but they had some significant differences as well. Attacks that affected widespread commodity hardware and software were thus not a major concern for aviation [15].

In recent years, both of these "defenses" have become significantly weaker or have been removed altogether, explaining an increased focus on aviation information systems security: more and more avionics systems are networked; they communicate with each other and some of them communicate with airline or other ground services, which in turn are connected to the Internet. Passenger entertainment and in-flight internet have become common, and they offer another vector of attack into avionics systems. In order to reduce costs, newer avionics systems are increasingly built using commercial off the shelf (COTS) hardware and software, which makes them cheaper to build and maintain, but also potentially vulnerable to common types of attacks carried out over the Internet. Using COTS hardware and software enables engineers to reuse code, tools, and parts, but it also enables attackers to do the same, thus dramatically reducing the cost of developing exploits against aviation. It is even imaginable that aircraft could be affected by attacks not specifically directed against them, such as a virus that spreads itself through a vulnerability in a common operating system.

2.4.1. Software Security

Considering the possible expectations people have about software security helps determine which issues they consider to be security violations. Security is often described as resting on three components: confidentiality, integrity, and availability (CIA) [17].

a) Confidentiality

Confidentiality requires that information be kept private. This includes any situation where software is expected to hide information or hide the existence of information. Software systems often deal with data that contains secrets, ranging from nation- or state-level intelligence secrets to company trade secrets or even sensitive personal information.

Software is often expected to compartmentalize information and ensure that only authenticated parties are allowed to see information for which they are authorized. These requirements mean that software is often expected to use access control technology to authenticate users and to check their authorization when accessing data. Encryption is also used to maintain the confidentiality of data when it is transferred or stored [17].

b) Integrity

Integrity is the trustworthiness and correctness of data. It refers to expectations that people have about software's capability to prevent data from being altered. Integrity refers not only to the contents of a piece of data, but also to the source of that data. Software can maintain integrity by preventing unauthorized changes to data sources.

Other software might detect changes to data integrity by making note of a change in a piece of data or an alteration of the data's origins.

Software integrity often involves compartmentalization of information, in which the software uses access control technology to authenticate users and check their authorization before they are allowed to modify data. Authentication is also an important component of software that is expected to preserve the integrity of the data's source because it tells the software definitively who the user is.

Typically, users hold similar expectations for integrity as they do for confidentiality. Any issue that allows attackers to modify information they would not otherwise be permitted to modify is considered a security flaw. Any issue that allows users to masquerade as other users and manipulate data is also considered a breach of data integrity. Software vulnerabilities can be particularly devastating in breaches of integrity, as the modification of data can often be leveraged to further an attackers' access into a software system and the computing resources that host the software [17].

c) Availability

Availability is the capability to use information and resources. Generally, it refers to expectations users have about a system's availability and its resilience to denial-of-service (DoS) attacks. An issue that allows users to easily crash or disrupt a piece of software would likely be considered a vulnerability that violates users' expectations of availability. This issue generally includes attacks that use specific inputs or environmental disruptions to disable a program as well as attacks centered on exhausting software system resources, such as central processing unit (CPU), disk, or network bandwidth [17].

2.4.2. Software Vulnerabilities

Software vulnerabilities are simply weaknesses in a system that attackers can leverage to their advantage. In the context of software security, vulnerabilities are specific flaws or oversights in a piece of software that allow attackers to do something malicious, expose or alter sensitive information, disrupt, or destroy a system, or take control of a computer system or program.

In general, software vulnerabilities can be classified into three: design vulnerability, implementation vulnerability and operation vulnerability. The security community generally accepts design vulnerabilities as flaws in a software system's architecture and specifications; implementation vulnerabilities are low-level technical flaws in the actual construction of a software system. The category of operational vulnerabilities addresses flaws that arise in deploying and configuring software in a particular environment.

A design vulnerability is a problem that arises from a fundamental mistake or oversight in the software's design. With a design flaw, the software is not secure because it does exactly what it was designed to do; it was simply designed to do the wrong thing! These types of flaws often occur because of assumptions made about the environment in which a program will run or the risk of exposure that program components will face in the actual production environment. Design flaws are also referred to as high-level vulnerabilities, architectural flaws, or problems with program requirements or constraints.

In an implementation vulnerability, the code is generally doing what it should, but there is a security problem in the way the operation is carried out. As it would be expected from the name, these issues occur during the implementation phase, but they often carry over into the integration and testing phase. These problems can happen if the implementation deviates from the design to solve technical discrepancies. Mostly, however, exploitable situations are caused by technical artifacts and nuances of the platform and language environment in which the software is constructed. Implementation vulnerabilities are also referred to as low-level flaws or technical flaws.

Operational vulnerabilities are security problems that arise through the operational procedures and general use of a piece of software in a specific environment. One way to distinguish these vulnerabilities is that they are not present in the source code of the software under consideration; rather, they are rooted in how the software interacts with its environment. Specifically, they can include issues with configuration of the software in its environment, issues with configuration of supporting software and computers, and issues caused by automated and manual processes that surround the system. In general, the label "operational vulnerabilities" is used for issues that deal with unsafe deployment and configuration of software, unsound management and administration practices surrounding software, issues with supporting components such as application and Web servers, and direct attacks on the software's users [17].

Although many software vulnerabilities result from processing malicious data, some software flaws occur when an attacker manipulates the software's underlying environment. These flaws can be thought of as vulnerabilities caused by assumptions made about the underlying environment in which the software is running. Each type of supporting technology a software system might rely on has many best practices and nuances, and if an application developer does not fully understand the potential security issues of each technology, making a mistake that creates a security exposure can be all too easy [17].

2.5. Current Regulations and Guidelines

To address the growing use of software in aviation, the Radio Technical Commission for Aeronautics (RTCA) released a document of guidelines for software development in 1982. The document was the result of consensus opinion from a wide range of domain experts periodically meeting as part of a working group tasked with the creation of guidelines. This document was titled DO-178, Software Considerations in Airborne Systems and Equipment Certification. Its purpose was to provide industry-accepted best practices to meet regulatory requirements for airworthiness such as Part 23 and Part 25 of the Federal Aviation Regulations (FARs). The document was revised in 1985 (DO-178A) and again in 1992 (DO-178B) to reflect the experience gained and criticisms of previous revisions. Although technically a guideline, it was a de facto standard for developing avionics software systems until it was replaced in 2012 by DO-178C.

It should be noted that DO-178 is not the only means of compliance with airworthiness regulations and that no portion of DO-178 is official government policy. RTCA is an association of government and private organizations that working together to develop recommended practices, but applicants for software approval are free to demonstrate compliance to the certification authority using any other acceptable means. Despite the possibility of developing software using alternative means of compliance, DO-178C is the de facto standard by which all software submissions for approval are judged at the local Aircraft Certification Offices (ACOs). The FAA approved AC 20-115C on 19-Jul-2013, which was superseded by AC 20-115D on 21-July-2017, making DO-178C a recognized "acceptable means, but not the only means, for showing compliance with the applicable airworthiness regulations for the software aspects of airborne systems and equipment certification".

While there is current FAA guidance and regulation pertaining to software development (DO-178C, AC 20-115C and Order 8110.49), new guidance for cyber-security (DO-326A and DO-355) has not yet been accepted or required as a means of compliance with airworthiness standards. ARINC Report 835 describes the digital signature-based process used by Boeing and Airbus to distribute software to their new aircraft [13].

2.5.1. RTCA DO-178C

RTCA DO-178C – "Software Considerations in Airborne Systems and Equipment Certification" outlines the process of developing software for aircraft. It also contains a section of guidance for the safety of field loadable software that reads as follows: "The safety-related requirements associated with the software data loading function are part of the system requirements. If the inadvertent enabling of the software data loading function could cause erroneous loading of software parts, then a safety-related requirement for the software data loading function should be specified in the system requirements" [12] [13].

System safety considerations relating to field-loadable software include:

- Detection of corrupted or partially loaded software
- Determination of the effects of loading the inappropriate software
- Hardware/software compatibility
- Software/software compatibility
- Aircraft/software compatibility
- Inadvertent enabling of the field loading function
- Loss or corruption of the software configuration identification display.

Unless otherwise justified by the system safety assessment process, the detection mechanism for partial or corrupted software loads should be assigned the same failure condition or software level as the most severe failure condition or software level associated with the function that uses the software load. If a system has a default mode when inappropriate software is loaded, then each partitioned component of the system should have safety-related requirements specified for operation in this mode which address the potential failure condition.

The software loading function, including support systems and procedures, should include a means to detect incorrect software and hardware, and should provide protection appropriate for the function involved. If the software consists of multiple configuration items, their compatibility should be ensured. If software is part of an airborne display mechanism that is the means for ensuring that the aircraft conforms to a certified configuration, then that software should either be developed to the highest software level of the software to be loaded, or the system safety assessment process should justify the integrity of an end-to-end check of the software configuration identification [12].

2.5.2. FAA Order 8110.49

FAA order 8110.49 is a guide for Aircraft Certification Service field offices and Designated Engineering Representatives (DERs) on how to apply RTCA DO-178B/C for approving software used in airborne computers. Chapter 5 of order 8110.49 concerns itself exclusively with the approval of field loadable software (FLS) and provides more detail than DO-178B/C [18].

The approved FLS may be installed on the aircraft via Service Bulletin, Engineering Change Request, or other FAA-approved means. The approved means vary, depending on the method for granting approval. Whether the FLS approval is through Type Certificate (TC), supplemental type certificates (STC), amended type certificates (ATC), amended supplemental type certificates (ASTC), technical standard order (TSO) authorizations (TSOA), or some other approval process, the document used to install the FLS should be approved by the certification authority and should specify the following elements [18]:

- The aircraft and hardware applicability and inter-mixability allowances for redundant systems software loading.

- Verification procedures to assure that the software was correctly loaded into an approved and compatible target computer and memory devices.
- Any post-load verification and/or test procedures required to show compliance to the guidelines specified in this chapter.
- Actions to be taken in the event of an unsuccessful load (for example, prohibit dispatch of the aircraft).
- Approved loading procedure or reference to approved loading procedure.
- Maintenance record entry procedures required to maintain configuration control.
- Reference to Aircraft Flight Manual, Aircraft Flight Manual Supplement, or Operator's Manual, as appropriate" [18].

2.5.3. RTCA DO-355

RTCA DO-355 – "Information Security Guidance for Continuing Airworthiness", prepared by RTCA SC-216 describes its purpose as follows: "This document provides guidance for the operation and maintenance of aircraft and for organizations and personnel involved in these tasks. It shall support the responsibilities of the Design Approval Holder (DAH) to obtain a valid airworthiness certificate and aircraft operators to maintain their aircraft to demonstrate that the effects on the safety of the aircraft of information security threats are confined within acceptable levels" [19].

DO-355 does not describe the process used for the distribution of software but refers to ARINC 667-1, ARINC 827 and ARINC 835 instead. It only provides guidance for the operational aspects of software distribution, such as the management of tools and media, the management of personnel and the handling of security incidents. It also describes means to ensure the security of network access points on the aircraft, including operational aspects of protecting the access points from unauthorized access.

2.5.4. RTCA DO-326A

RTCA DO-326A – "Airworthiness Security Process Specification", prepared by RTCA SC-216 and EUROCAE WG-72 aims to specify an acceptable means to demonstrate security of an aircraft system for the purposes of certification: "This guidance material is for equipment manufacturers, aircraft manufacturers, and anyone else who is applying for an initial Type Certificate (TC), and afterwards for Supplemental Type Certificate (STC), Amended Type Certificate (ATC) or changes to Type Certification for installation and continued airworthiness for aircraft systems" [20].

The purpose of the Airworthiness Security Process (AWSP) is to establish that, when subjected to unauthorized interaction, the aircraft will remain in a condition for safe operation (using the regulatory airworthiness criteria). To accomplish this purpose, the Airworthiness Security Process:

- Establishes that the security risk to the aircraft and its systems are acceptable per the criteria established by the AWSP, and
- Establishes that the Airworthiness Security Risk Assessment is complete and correct.

As part of the security process, DO-326A requires defining a security scope, performing various security risk assessments, designing a security architecture, and finally performing a security verification. Each step is described in sufficient detail so it could be implemented by a supplier, manufacturer, or an airline.

2.5.5. FAA AC 20-115D

This advisory circular, AC 20-115D – “Airborne Software Development Assurance”, describes an acceptable means, but not the only means, for showing compliance with the applicable airworthiness regulations for the software aspects of airborne systems and equipment in type certification or TSO authorization [21]. It recognizes RTCA DO-178C and RTCA DO-33( ) supplements to DO-178C. This AC applies to design approval holders, and developers of airborne systems and equipment containing software to be installed on type certificated aircraft, engines, and propellers, or to be used in TSO articles.

2.5.6. ARINC Report 835

This report is published in 2014 by Aeronautical Radio Inc (ARINC) to provide background and detailed technical information on existing methods to secure loadable software parts [22]. It describes the standards used by aircraft manufacturers (e.g., Boeing & Airbus) for applying digital signatures to field loadable software for their new aircraft types, such as B787 and A350. It is intended as guidance for other organizations who wish to implement digital signature schemes. The two main purposes of digital signatures for field loadable software parts that the report describes are integrity and authentication independent from the media used for transporting the software (network, portable media, etc.). Non-repudiation is mentioned as a secondary goal. The schemes used by both Boeing and Airbus rely on a Public Key Infrastructure (PKI) and add a header file to the software being signed, but they differ in the type and number of signatures added to the software.

2.5.7. Summary of Current Regulations and Guidelines

The review of the current regulations and guidelines related to the security of aircraft software indicates that the focus is given mainly to the development and certification of the software. The purpose of these regulations and guidelines is basically to make sure that only approved and certified software parts that showing compliance with the applicable airworthiness regulations are installed on the aircraft.

The security of field loadable software parts distribution is described in ARINC Report 835, but it is applied to the latest e-enabled aircraft types that adapted technological advances such as Internet Protocol (IP) connectivity and that require "Special Condition" as defined in AC 119-1 [28]. This advanced technology can be found only on the new aircraft designs and does not apply to the previous aircraft designs that utilized ARINC 429 and ARINC 629 data buses to connect flight-critical avionics systems [28]. The aircraft manufacturers and software suppliers use the ARINC Report 835 guidelines and standards for applying digital signatures to the software parts as a security scheme [15].

Advisory Circular AC 43-216 states that "Operators should have a software management process that includes adequate protections from software tampering while the software is in storage and during transfers. This is required for all aircraft software" [11]. "There are various methods of receiving, storing, transporting, and loading of software" [11]. Therefore, it does not specify which cybersecurity mechanisms and/or architectures should be used. It is the responsibility of airlines to perform security assessment and utilize the security processes and mechanisms that can provide adequate security protections.

2.6. Related Works

This section presents a review of some selected related works on Cybersecurity Mechanism for Aircraft Systems' Software Distribution System. The gaps that exist in previous works are analyzed, identified, and summarized.

a) Richard Robinson et al., "Electronic Distribution of Airplane Software and the Impact of Information Security on Airplane Safety" [10]:

The objective of this study is to identify threats to Airplane Assets Distribution System (AADS), and to analyze and address threats against the integrity of software of highest criticality. Their approach is based on the Common Criteria (CC) methodology and Information Assurance Technical Framework (IATF), respectively. CC is a framework in which computer system users can specify their security functional and assurance requirements in a security target. Whereas IATF provides guidance for protecting the information infrastructures through information assurance that addresses all the security requirements. The researchers proposed using digital signatures to secure distributed assets in the AADS. The major limitation of this study is that it is applicable only to the latest e-enable airplane that adapted technological advances such as IP, as indicated in AC 119-1, for end-to-end electronic software distribution. It does not cover airplanes with previous designs and software parts that are distributed through physical transport media such as flash disks, compact disks, etc.

b) David von Oheimb et al., "Security Architecture and Formal Analysis of an Airplane Software Distribution System" [14]:

The objective of this study is to introduce a software distribution system architecture with a generic core component, such that the overall software transport from the supplier to the airplane is an interaction of several instances of this core component communicating over insecure networks. They outlined the security evaluation of the proposed system according to the Common Criteria (CC) methodology. The aim of evaluation according to the CC was to systematically and objectively demonstrate that countermeasures are sufficient and correctly implemented. The researchers simplified the system design by defining a generic core component called the Asset Signer Verifier (ASV) to be used at every node of the system and they proposed digital signatures as security mechanism. Because of this, it can be applied to both latest and previous airplane designs. However, since it makes each ASV responsible for applying digital signatures on software items before transmitting them and verifying signatures when received from other entities, this could be relatively costly and impractical for some owners/operators as their existing IT infrastructure may not support it. The other limitation of this study is similar to that of [10] that the approach applies to electronically distributed software parts.

c) Jonas Helfer, "A Systems Approach to Software Security in Aviation" [15]:

The aim of this thesis is to provide a comprehensive security analysis of Field Loadable Software that includes organizational aspects in order to find existing vulnerabilities and propose security constraints that would fix the vulnerabilities or prevent them from being exploited. It adopted Systems Theoretic Process Analysis (STPA) to perform the security analysis. The researcher produced hierarchical control structure to determine which control actions are potentially unsafe. This study recommended that simple changes, such as the addition of a hardware supported cryptographic hash for software components, and minor modifications to the loading process could be used to control the biggest hazards and eliminate most loss scenarios. The limitation of this study is that it focuses on the organizational aspects rather than technical aspects. It was entirely based on recommended practices and regulations, and it did not study any particular supplier or airline's practices and did not show how it could be applied to each phase of the software distribution.

2.7. Gap Analysis

The software currently used in commercial aircraft has to adhere to numerous standards intended to ensure safety, and the industry has managed to significantly reduce accident rates over the last 50 years [23]. However, the security of aviation software has only recently started to gain attention, and is thus not very well understood [10, 24].

According to FAA's Strategic Plan for fiscal years 2019 through 2022, the agency has adopted NIST's Cybersecurity Framework to reduce aviation critical infrastructure risk. However, FAA has not assessed the risks to avionics systems to determine the relative priority of cybersecurity risks to avionics systems versus other safety concerns in its oversight program. While the agency uses its Safety Management System to assess risks for certification projects, this form of risk assessment is only at the certification project-level and, therefore, does not inform a larger agency strategy to oversee industry actions to address avionics cybersecurity risks [6].

As airplanes and aircraft systems are increasingly reliant on complex software to perform their functions, software parts and their distribution from end-to-end should be equipped with better security platforms. The current regulations and guidelines leave the responsibility to the airlines to adequately protect the software parts while they are in storage and during transfers, without specifying the mechanisms need to be implemented.

The review of previous studies showed that the security analysis of LSAP distribution is not comprehensive. Most of the previous security analysis are based on the Common Criteria (CC) methodology. The CC does not contain security evaluation criteria pertaining to administrative security measures not related directly to the IT security functionality [25]. Also, the previous studies did not take the existing airline distribution infrastructure into account and did not address the security risks for all aircraft design types, i.e., they covered the latest e-enabled aircraft types that adapted technological advances, but they cover the previous aircraft design types. Based on these facts, we can say that the previous works on the security of aircraft software distribution system did not cover the overall security issues.

Currently, unlike the latest design airplanes that use digital signatures, there is no security mechanism that can prevent installation of software from non-official sources for the previous design airplanes [28] [6]. The installed software might have been tampered with by an attacker before installation, without changing the software version number or falsifying it as an official version. In addition to the aforementioned issue, the access to the software data distribution and storage servers and maintenance laptops is controlled by using single factor authentication, i.e., username and password only. Nowadays, a single factor authentication is no longer considered secure because it can easily be compromised, and bad actor can take advantage to tamper the software [40] [42].

Therefore, this research addresses these gaps by carrying out a comprehensive security assessment using Systems Approach called Systems Theoretic Process Analysis for Security (STPA-Sec) to identify potential security vulnerabilities and applying a multi-layer security mechanism that will enhance the existing security mechanisms. The reason for selecting the STPA-Sec is it differs from most other security analysis methods in that it uses a strategic approach instead of a tactical approach. By forcing the analysts to think about the high-level system goals, the method makes it more likely that simpler and more radical solutions will be found where they are possible. The tactical approaches ask the analyst to identify and fix vulnerabilities. Such approaches are more likely to lead to a patchwork of fixes, often with the ultimate result that security is only enhanced until the next vulnerability is found. The proposed security mechanism is used to design the artifact, i.e., a new security architecture, and the artifact is evaluated to ensure that the identified security issues have been properly addressed.

2.8. Systems Theoretic Process Analysis (STPA)

Systems Theoretic Process Analysis (STPA) is a powerful hazard analysis technique based on the Systems Theoretic Accident Model and Processes (STAMP) model that has proven to be a powerful tool to study emergent properties of systems such as safety and security [30]. STAMP is an accident causality model based on systems theory and systems thinking, which integrates into engineering analysis causal factors such as software, human decision-making and human factors, new technology, social and organizational design, and safety culture, which are becoming ever more threatening in our increasingly complex systems. Application areas of this technique includes aviation, air traffic control, space, defense, the automotive industry, railways, chemicals, oil and gas, medical devices, healthcare, and workplace safety, with a growing interest coming from new areas such as the pharmaceutical industry and the finance and insurance sectors.

STAMP is a new model for accidents that is based on control theory. Under this model, safety is considered a control problem: Accidents occur when the system does not adequately control the process it is built to control. This can occur for a number of reasons, such as: component failures, component interaction and events outside of the range the system was designed to handle. The goal of an STPA analysis is therefore to find scenarios in which the system is unable to keep the process from entering hazardous states in which accidents can occur [15] [30].

2.8.1. STPA in Practice

The initial step when performing an STPA analysis is to define the system losses and the hazards that can lead to these losses. Usually, these are defined at high level. As a rule of thumb, there are between 2 and 5 system-level losses and less than 10 high level system hazards leading to the system losses [15] [30].

It is important that the hazards are actually under the system's control. If the hazards cannot be controlled by the system, then either the model of the system needs to be re-scoped and its boundary expanded, or the hazards need to be redefined. On the other hand, if only a part of the system is necessary to control the hazards, the system boundary may have been drawn too wide.

Once the losses and hazards are determined, a hierarchical control structure is drawn that includes all the controllers and control actions in the system that are necessary to keep the controlled process out of its hazardous states. Controllers are arranged in a hierarchy, such that the controlled process is at the bottom. The hierarchical control structure serves to illustrate the system architecture. For every controller, each control action needs to be studied to determine whether it could potentially be an unsafe control action.

Once the table or list of unsafe control actions for each controller is established, the next step of STPA consists in finding scenarios that could lead to these unsafe control actions in order to propose safety constraints that will keep these scenarios from happening. Finding such safety constraints should be the final output of an STPA analysis, and no analysis is complete without them. Scenarios should be generated in a methodical way by people with expert knowledge in the system. A control loop diagram for each controller can be used to assist in the process.

2.8.2. STPA-Sec

STPA for security (STPA-Sec) is an extension to STPA that adds the following elements to the classical STPA analysis [15] [30]:

- The realization that STPA can be applied to security equally well as safety.
- Producing a high-level system description becomes an explicit first step in the analysis. The high-level system description includes answering the following questions: What does the system do? How does the system do it? Why does the system do what it does?
- Before listing losses and hazards, a “mission” for the system is explicitly written down. The mission defines the initial state of the system, the end state of the system as well as all intermediate states necessary to attain the end state. The mission can then be divided up into stages, each of which is supposed to bring the system from one state to the next.

STPA-Sec differs from most other security analysis methods in that it uses a strategic approach instead of a tactical approach. By forcing the analyst to think about the high-level system goals, the method makes it more likely that simpler and more radical solutions will be found where they are possible. Contrast this with more tactical approaches which ask the analyst to identify and fix vulnerabilities. Such approaches are more likely to lead to a patchwork of fixes, often with the ultimate result that security is only enhanced until the next vulnerability is found.

2.8.3. STPA-Sec Guidelines

There are four basic steps used to perform the Systems Theoretic Process Analysis for Security (STPA-Sec analysis) [30] [31]. These are:

(1) Define purpose of the analysis

(2) Model the control structure

(3) Identify unsafe control actions and

(4) Identify loss scenarios

Figure 2 shows the basic STPA steps along with their graphical representations and the detail description of each step is provided below.

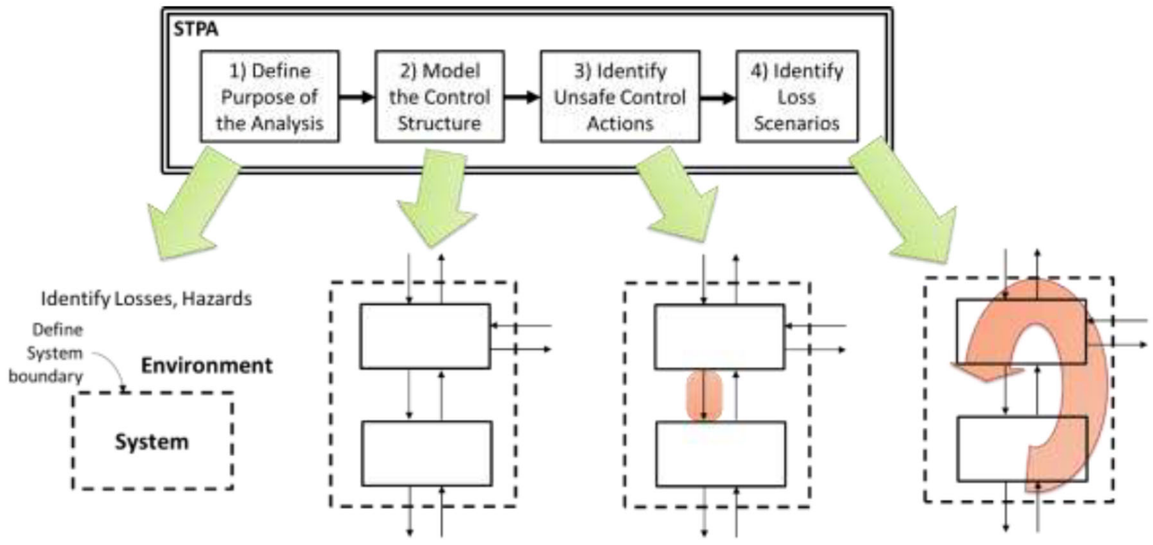
STPA
1) Define Purpose of the Analysis
2) Model the Control Structure
3) Identify Unsafe Control Actions
4) Identify Loss Scenarios
Identify Losses, Hazards
Define System boundary
Environment
System

Figure 2. Overview of the basic STPA Method

Source: [30] [31]

1) Define Purpose of the Analysis:

Defining the purpose of the analysis is the first step with any analysis method. What kinds of losses will the analysis aim to prevent? What is the system to be analyzed and what is the system boundary? These and other fundamental questions are addressed during this step.

Defining the purpose of the analysis has four parts: (a) identifying losses, (b) identifying system-level hazards, (c) identify system-level constraints, and (d) refine hazards.

(a) identifying losses

A loss involves something of value to stakeholders. Losses may include a loss of human life or human injury, property damage, loss or leak of sensitive information, loss of mission, loss of reputation, environmental pollution, or any other loss that is unacceptable to the stakeholders.

(b) identifying system-level hazards

A hazard is a system state or set of conditions that, together with a particular set of worst-case environmental conditions, will lead to a loss. Whereas A system is a set of components that act together as a whole to achieve some common goal, objective, or end. A system may contain subsystems and may also be part of a larger system.

(c) identify system-level constraints

A system-level constraint specifies system conditions or behaviors that need to be satisfied to prevent hazards (and ultimately prevent losses).

(d) refine hazards

Once the list of system-level hazards has been identified and reviewed, these hazards can be refined into sub-hazards if appropriate. The first step in refining the system-level hazards is to identify basic system processes or activities that need to be controlled to prevent system hazards.

2) Model the Control Structure:

The second step is to build a model of the system called a control structure. A control structure captures functional relationships and interactions by modeling the system as a set of feedback control loops. The control structure usually begins at a very abstract level and is iteratively refined to capture more detail about the system. This step does not change regardless of whether STPA is being applied to safety, security, privacy, or other properties.

In general, a controller may provide control actions to control some process and to enforce constraints on the behavior of the controlled process. The control algorithm represents the controller's decision-making process - it determines the control actions to provide. Controllers also have process models that represent the controller's internal beliefs used to make decisions. Process models may include beliefs about the process being controlled or other relevant aspects of the system or the environment. Process models may be updated in part by feedback used to observe the controlled process.

3) Identify Unsafe Control Actions:

The third step is to analyze control actions in the control structure to examine how they could lead to the losses defined in the first step. Unsafe Control Actions (UCAs) are control actions that, in a particular context and worst-case environment, will lead to a hazard. These unsafe control actions are used to create functional requirements and constraints for the system. This step also does not change regardless of whether STPA is being applied to safety, security, privacy, or other properties.

There are four ways a control action can be unsafe: (a) not providing the control action leads to a hazard, (b) providing the control action leads to a hazard, (c) providing a potentially safe control action but too early, too late, or in the wrong order, and (d) the control action lasts too long or is stopped too soon (for continuous control actions, not discrete ones).

4) Identify Loss Scenarios:

The fourth step identifies the reasons why unsafe control might occur in the system. Loss scenarios describe the causal factors that can lead to the unsafe control actions and to hazards. Scenarios are created to explain: (a) How incorrect feedback, inadequate requirements, design errors, component failures, and other factors could cause unsafe control actions and ultimately lead to losses. (b) How safe control actions might be provided but not followed or executed properly, leading to a loss.

Once scenarios are identified, they can be used to create additional requirements, identify mitigations, drive the architecture, make design recommendations and new design decisions (if STPA is used during development), evaluate/revisit existing design decisions and identify gaps (if STPA is used after the design is finished), define test cases and create test plans, and develop leading indicators of risk.

CHAPTER THREE -
RESEARCH METHODOLOGY

3.1. Overview

This chapter discusses the processes and techniques used in carrying out the research. More specifically, it presents how the study is systematically designed to ensure valid and reliable results that address the research aims and objectives.

This study employs design science research methodology. Design science research is an outcome-based information technology research methodology. It focuses on the development and performance of (designed) artifacts with the explicit intention of improving the functional performance of the artifact. Design science research involves a rigorous process to design artifacts to solve observed problems, to make research contributions, to evaluate the designs, and to communicate the results to appropriate audiences [26].

3.2. Design Science Research

Design science research method provides a framework for creating new theories and artifacts that could be constructs, models, frameworks, architectures, design principles, methods, instantiations, and design theories. Design science research is made of three pillars: environment, information system research and knowledge base. Figure 3 illustrates the conceptual framework for understanding, executing, and evaluating design science research. The environment represents the people and technology that make up an organization. While the knowledge base represents previous knowledge foundations such as theories, methods, models, technique, measures, etc. Design science helps organizations to address their business needs by conducting research that are relevant to the problems of the organizations. The research is also conducted rigorously by applying knowledge from the knowledge base to develop new artifacts and also to evaluate such artifacts [27].

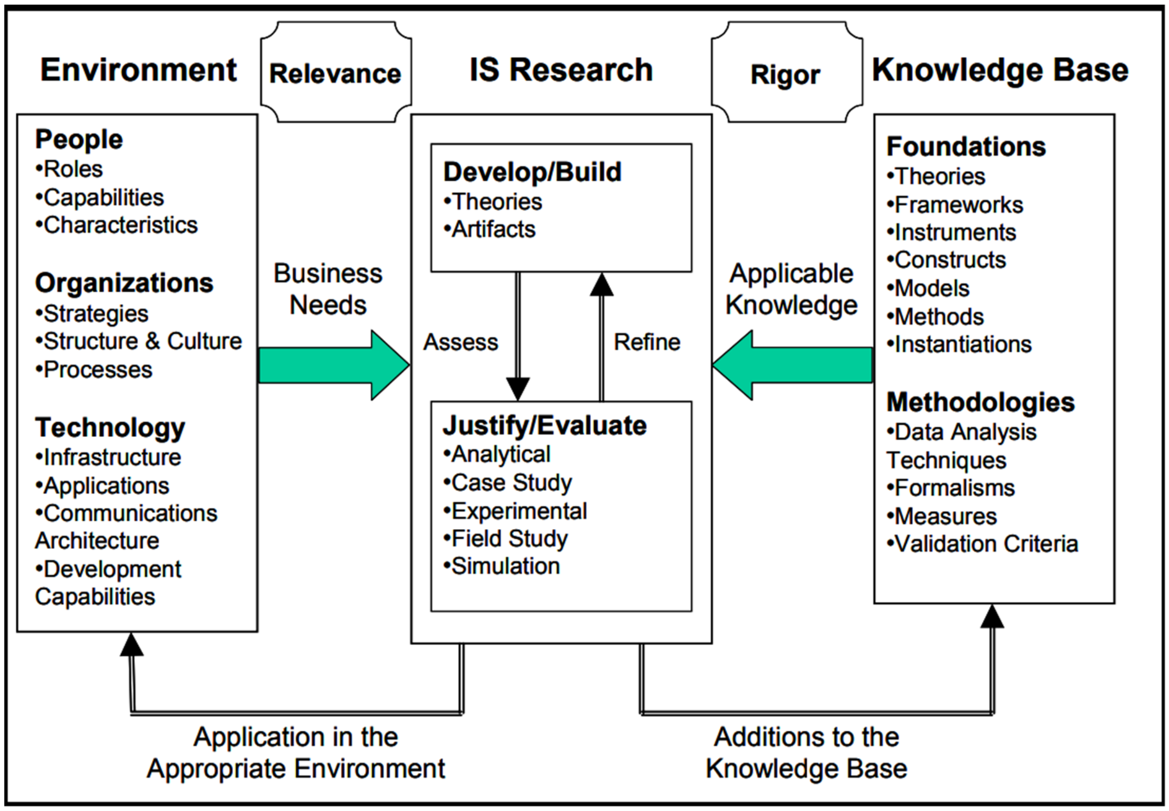

Figure 3. Design science research framework for the domain of information systems
Source: [27]

Design science is inherently a problem-solving process. The fundamental principle of design science research from which the six guidelines are derived is that knowledge and understanding of a design problem and its solution are acquired in the building and application of an artifact. That is, design science research requires the creation of an innovative, purposeful artifact for a specified problem domain. Because the artifact is purposeful, it must yield utility for the specified problem. Hence, thorough evaluation of the artifact is crucial. Novelty is similarly crucial since the artifact must be innovative, solving a heretofore unsolved problem or solving a known problem in a more effective or efficient manner. The artifact itself must be rigorously defined, formally represented, coherent, and internally consistent. The process by which it is created, and often the artifact itself, incorporates or enables a search process whereby a problem space is constructed, and a mechanism posed or enacted to find an effective solution. Finally, the results of the design science research must be communicated effectively [27].

3.3. Research Design

Research designs are plans and the procedures for research that span the decisions from broad assumptions to detailed methods of data collection and analysis [16]. Research design is the framework for a study. Designing a research study has often been compared to designing a building. There are a number of reasons why it is a good idea to plan a house before actually building it. Like building plans, research designs ensure that the study fulfils a particular purpose, and the research can be completed with available resources.

3.3.1. Design Science Research Guidelines

Design Science Research process generally includes six steps or activities [26]: (1) identification of the problem, defining the research problem and justifying the value of a solution; (2) definition of objectives for a solution; (3) design and development of artefacts (constructs, models, methods, etc.); (4) demonstration by using the artefact to solve the problem; (5) evaluation of the solution, comparing the objectives and the actual observed results from the use of the artefact; and (6) communication of the problem, the artefact, its utility and effectiveness to other researches and practicing professionals.

Figure 4 presents a process model consisting of six activities in a nominal sequence.

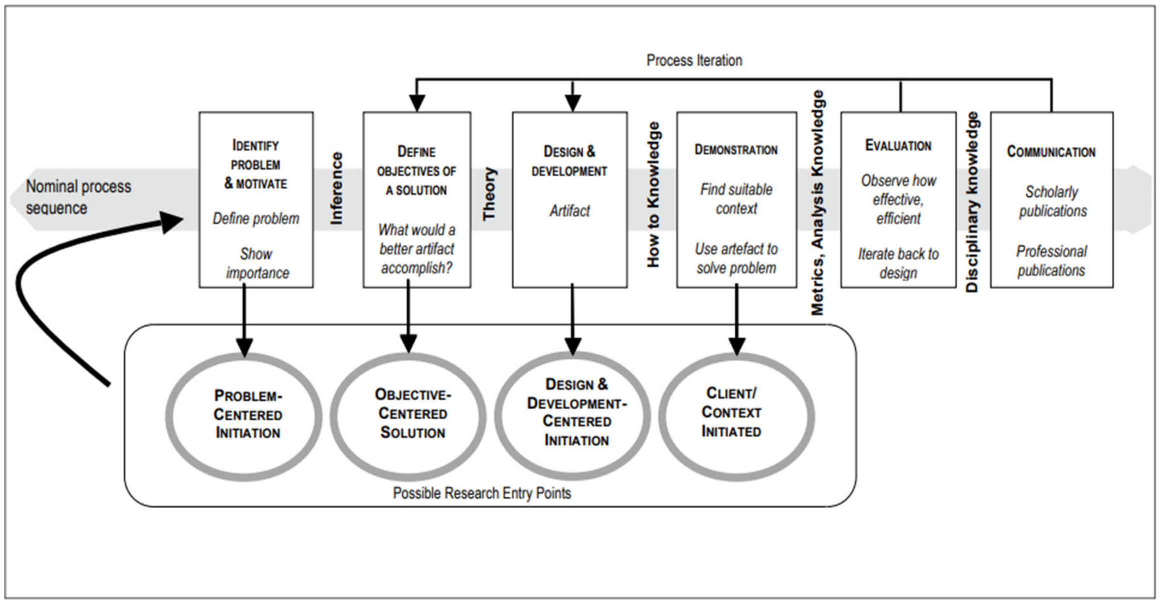
Process Iteration
Nominal process sequence
IDENTIFY PROBLEM & MOTIVATE
Define problem
Show importance
Inference
DEFINE OBJECTIVES OF A SOLUTION
What would a better artifact accomplish?
Theory
DESIGN & DEVELOPMENT
Artifact
How to Knowledge
DEMONSTRATION
Find suitable context
Use artefact to solve problem
Metrics, Analysis Knowledge
EVALUATION
Observe how effective, efficient
Iterate back to design
Disciplinary knowledge
COMMUNICATION
Scholarly publications
Professional publications
PROBLEM-CENTERED INITIATION
OBJECTIVE-CENTERED SOLUTION
DESIGN & DEVELOPMENT-CENTERED INITIATION
CLIENT/CONTEXT INITIATED
Possible Research Entry Points

Figure 4. Design Science Research Methodology Process Model
Source: [26]

The following describes how this research is conducted according to the guidelines of the design science framework:

1. **Identification of the Problem**

The current aviation regulations and guidelines do not specifically address the issue of software distribution and its security. It is the airlines responsibility to assess their system and apply a software management process that provide adequate security protections. Airlines utilize various security mechanisms in various phases of software distribution and various security architectures for different types of the airplanes and aircraft designs. In addition to this, review of the previous studies showed that the security analysis of avionics systems software distribution was not comprehensive, and it only covered the latest aircraft designs, not taking the previous aircraft designs into account [10,14,15]. Thus, there is a need to assess the software distribution system and its inherent security in order to develop an efficient security architecture and apply multi-layer security mechanism for adequate protection.

2. Definition of Objectives for Solution

The general objective in this research work is to develop an efficient security architecture by applying multi-layer security mechanism for adequate protection. The specific objectives are: (1) to analyze the cybersecurity mechanisms that are currently applied to the aircraft software distribution system for potential vulnerabilities. (2) to design an artifact, i.e., security architecture, using the proposed multi-layer security mechanism. (3) to evaluate the designed architecture to ensure the identified security issues have been properly addressed.

3. Design and development of artifacts

The artifact is a security architecture that applies multi-layer security mechanism and efficiently utilizes resources. This study adopts Systems Theoretic Process Analysis for Security (STPA-Sec) method to carry out the comprehensive security analysis of the aircraft software distribution system and to identify potential security vulnerabilities. In addition to the commercial aviation general data, the researcher's observations, and experience as an engineering professional are used as an input for the analysis of the company's aircraft software distribution system as well as its security. The findings of this analysis contribute to the design of the artifact, i.e., design of the security architecture for the LSAP distribution system. The security architecture is designed by applying the multi-layer security mechanisms and created using network diagrams.

4. Demonstration

After designing and developing the artifact (security architecture), we demonstrate efficacy and efficiency of the artifact to the subject matter experts (i.e., avionics engineers and IT professionals) in the airlines industry and it was focused on how the proposed architecture addresses the security problems. The artifact addresses the existing security vulnerabilities by applying the multi-layer security mechanisms. It also streamlines and enhances the efficiency of the existing security architecture. The artifact is also demonstrated to the university academic researchers and evaluators.

5. Evaluation

The designed security architecture meets the research objectives, which was evaluated using the Cyber Security Evaluation Tool (CSET). It was developed by cybersecurity experts under the direction of Cybersecurity and Infrastructure Security

Agency (CISA). The tool provides users with a systematic and repeatable approach to assessing/evaluating the security posture of their cyber systems and networks. It includes both high-level and detailed questions related to all industrial control and IT systems [39].

The evaluation is carried out using the following steps [39]: (a) Select Standards: select one or more government and industry recognized cybersecurity standards. (b) Determine Security Assurance Level (SAL): it is determined by responses to questions relating to the potential consequences of a successful cyber-attack on a critical system organization, facility, system, or subsystem. (c) Create Diagram: it contains a graphical user interface that allows users to diagram network topology and identify the "criticality" of the network components. (d) Answer Questions: select the best answer to each question using the organization's actual network configuration and implemented security policies and procedures. (e) Review Analysis and Reports: the analysis dashboard provides interaction with graphs and tables that present the assessment results in both summary and detailed form.

6. Communication

This thesis is written for partial fulfilment of the requirements for the Degree of Master of Science in Computer Science. The artifact of this study could be helpful for airlines as it streamlines the software distribution system and it could also contribute to technical audiences such as aviation experts, cybersecurity experts, etc., as an input for further study. The paper will be available for the research people and the university community.

CHAPTER FOUR - SECURITY ANALYSIS OF LSAP DISTRIBUTION

4.1. Overview

This chapter presents the comprehensive security analysis of the aircraft software distribution system using the systems approach called Systems Theoretic Process Analysis (STPA). This chapter will also provide an overview of the STPA-Sec and then apply it to carry out the security analysis of the LSAP distribution system in the airlines.

Some advantages of using STPA-Sec are: (a) it works on complex systems because it works top-down rather than bottom up, (b) it uses a strategic approach forcing to think about the high-level system goals, which helps to find simpler and more radical solutions, (c) it includes software, humans, organizations, safety culture, etc. as causal factors in accidents and other types of losses [30].

4.2. STPA-Sec Analysis of LSAP Distribution System

This section presents the steps and results of the STPA analysis performed for the security of Field Loadable Software processes in commercial aviation. The analysis in this section follows the structure of STPA-Sec, as discussed in chapter 2 and 3.

4.2.1. System Purpose

The purpose of LSAP processes (what the system does) is to *keep LSAP safe and secure*; by means of (how the system does what it does): *(1) developing safe and secure software and updates, (2) distributing and installing only safe and secure software on all applicable aircraft, and (3) monitoring LSAP in operation for flaws and attempted manipulation*; in order to contribute (why the system does what it does) to *safe and efficient commercial aviation [11] [43]*.

4.2.2. Losses and Hazards

Accident (Loss) is an undesired or unplanned event that results in a loss, including loss of human life or human injury, property damage, loss or leak of sensitive information, loss of mission, loss of reputation, environmental pollution, mission loss, etc. Whereas Hazard is a system state or set of conditions that, together with a particular set of worst-case environment conditions, will lead to an accident (loss). For the LSAP distribution system, the losses and hazards are identified as follows:

Losses:

L1: *Death or injury to pilot, passengers, or bystanders*

L2: *Damage to aircraft or equipment*

Hazards:

H1: *Loss of control of aircraft (aerodynamic/structural)* → [L1, L2]

H2: *Incorrect reporting (or unable to maintain safe operation)* → [L1, L2]

Secondary Hazards (concern LSAP only):

H3: *Unsafe or insecure software installed on the aircraft* → [H1, H2]

H4: *Software is installed improperly on the aircraft* → [H1, H2]

4.2.3. Control Structure

Generally, the commercial aviation has nine controllers: FAA, foreign regulatory agency, software supplier, airline management, airline mechanic, MRO (Maintenance, Repair and Overhaul) management, MRO mechanics, pilots, and LSAP. This is due to mostly airlines/operators perform light aircraft maintenance by themselves and outsource heavy aircraft maintenance to third party MRO organizations.

However, in the case of the airlines in question that this study focused all of aircraft maintenance is performed by their own MRO and they do not outsource to third party MROs. Hence, these airlines operations have the following seven controllers (i.e., FAA/ EASA, CAA, manufacturer/ supplier, airline/ MRO management, airline/ MRO mechanics, and pilots) and associated system-level constraints:

FAA (Federal Aviation Administration)/ EASA:

- Ensures that only safe and secure software and aircraft configurations get FAA/ EASA (European Union Aviation Safety Agency) type certificates.
- Issues Airworthiness Directive (AD) for any safety-critical flaws in LSAP that have a type certificate.
- Ensures that airline operations follow FAA/ EASA regulations.

CAA (Civil Aviation Authority):

- Ensures that airlines/MROs are certified only in accordance with regulations and bilateral agreement/ treaty.
- Ensures that airline operations follow CAA/ FAA/ EASA regulations.
- Certifies mechanics and pilots.

Manufacturer(s)/ Software Supplier(s):

- Develop safe and secure field loadable software and accompanying manuals.
- Distribute software parts and updates to airlines.
- Deliver service bulletins to FAA/ EASA and all airlines affected by a flaw.

Airline/ MRO Management (including Engineering, IT, Planning, etc.):

- Provides manuals and training to mechanics and pilots.
- Ensures software management process is secure and safe to store and distribute software parts and updates to Airline/ MRO mechanics.
- Ensures pilots are qualified & fit to fly, and mechanics are qualified to maintain.
- Ensures that aircraft stays airworthy by maintaining aircraft regularly.
- Ensures that pilot's work schedule is in accordance with local regulation.
- Ensures that mechanics carry out work in accordance with local regulation.

Airline/ MRO Mechanics:

- Install and update LSAP on aircraft in accordance with instructions provided by the Airline/ MRO management.
- Make sure aircraft is airworthy before takeoff.

Pilots:

- Check configuration before takeoff and follow warnings.

- Operate aircraft safely.

LSAP (Software):

- Issue warning when aircraft is in danger, e.g., flying into terrain or obstacles.

The control structure in Figure 5 shows the basic controllers in airlines. Almost all of the maintenance is carried out by the airline itself, i.e., the airline has its own Maintenance Repair and Overhaul (MRO) organization to carry out its own maintenance as well as other third-party airlines' maintenance. The airline and its MRO are both certified and regulated by CAA with whom the FAA/ EASA has a bilateral agreement. The FAA/ EASA can inspect these facilities, but only after coordinating with the corresponding regulatory agency (CAA) and the corresponding government. Technically the agreements are between governments and not between the FAA/ EASA and another regulatory agency, but for the purposes of this figure it was simpler to draw a direct link without compromising the analysis.

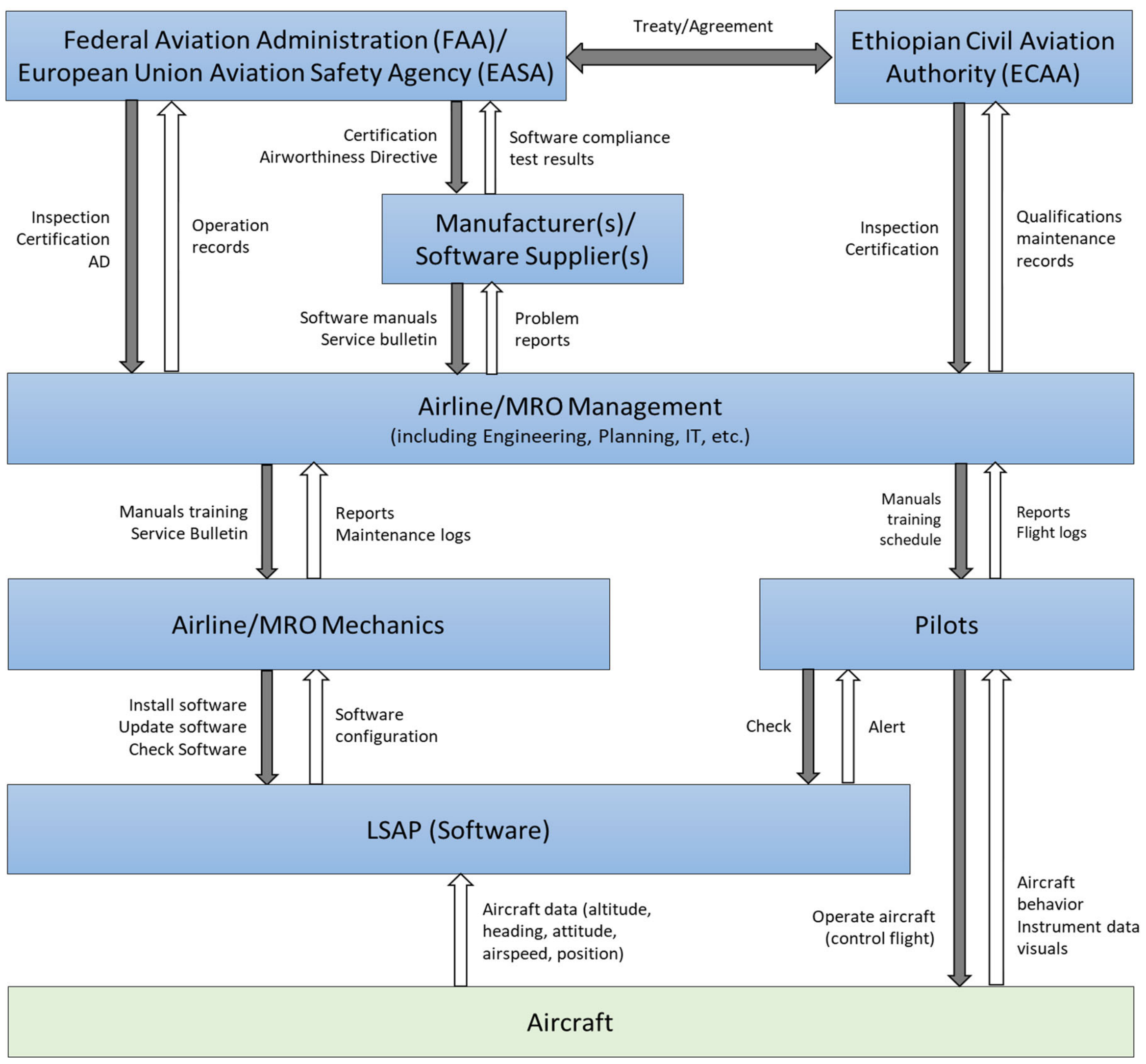
Federal Aviation Administration (FAA)/ European Union Aviation Safety Agency (EASA)
Treaty/Agreement
Ethiopian Civil Aviation Authority (ECAA)
Inspection Certification AD
Operation records
Certification Airworthiness Directive
Software compliance test results
Manufacturer(s)/ Software Supplier(s)
Inspection Certification
Qualifications maintenance records
Software manuals Service bulletin
Problem reports
Airline/MRO Management
(including Engineering, Planning, IT, etc.)
Manuals training Service Bulletin
Reports Maintenance logs
Manuals training schedule
Reports Flight logs
Airline/MRO Mechanics
Pilots
Install software Update software Check Software
Software configuration
Check
Alert
LSAP (Software)
Aircraft data (altitude, heading, attitude, airspeed, position)
Operate aircraft (control flight)
Aircraft behavior Instrument data visuals
Aircraft

Figure 5. Control Structure for LSAP in airlines

(This figure documents the existing control structure for STPA-Sec analysis)

4.2.4. Unsafe Control Actions (UCAs)

Once the control structure has been modeled, the next step is to identify unsafe control actions. Unsafe control action is a control action that, in particular context and worst-case environment, will lead to a hazard. For aircraft software parts, there are three ways a control action can be unsafe: (1) providing the control action leads to a hazard, (2) not providing the control action leads to a hazard, and (3) providing a potentially safe control action but at the wrong time (i.e., too early or too late). In STPA-Sec method, unsafe control actions are represented using a matrix of the control actions versus cases that control actions can be unsafe.

This section lists the unsafe control actions (UCAs) for each controller and provides additional explanations where necessary.

1) FAA/ EASA UCAs

The FAA/ EASA is responsible for overseeing aircraft manufacturer, software supplier(s), and airline/ MRO operations (if the organizations fall under its jurisdiction). It should coordinate with foreign regulatory agencies (e.g., CAA) for oversight of airlines/ MROs that do not fall under its jurisdiction. The FAA/ EASA must ensure that it does not issue type certificates for LSAP configurations that are unsafe or insecure. If an update to LSAP is necessary, the FAA/ EASA must issue an airworthiness directive (AD) quickly. Furthermore, the FAA/ EASA is responsible for licensing airlines. Airlines that do not follow safe maintenance practices with respect to LSAP should not be licensed.

Table 1 shows the FAA/ EASA unsafe control actions (UCAs). For FAA/ EASA there are seven UCAs (UCA-1.1 thru UCA-1.7). For example, let us consider UCA-1.1: Provides initial type certificate when LSAP software is unsafe or insecure => [H3, H4]. This UCA is unsafe because it can lead to hazards H3 (*Unsafe or insecure software installed on the aircraft*) and H4 (*Software is installed improperly on the aircraft*), which are defined in section 4.2.2 (Losses and Hazards). In turn, hazards H3 and H4 can lead to losses L1 (*Death or injury to people*) and L2 (*Damage to aircraft/ equipment*), as defined in section 4.2.2. Thus, if FAA/ EASA provides type certificate when LSAP software is unsafe or insecure, that will lead to death or injury to people and damage to equipment. Similar explanation works for UCA-1.2 thru UCA-1.7.

Table 1. FAA/ EASA Unsafe Control Actions

Control Action	**Providing causes hazard**	**Not providing causes hazard**	**Too soon/ too late caused hazard**
Provide initial type certificate	UCA-1.1: Provide type certificate when LSAP software is unsafe or insecure ➔ [H3, H4]	-	-
Provide type certificate for update	UCA-1.2: Provide type certificate when LSAP software is unsafe or insecure ➔ [H3, H4]	UCA-1.3: Type certificate for update is not provided when current LSAP is hazardous and unsafe or insecure ➔ [H3, H4]	UCA-1.4: Certificate for update is provided too late when current LSAP is unsafe or insecure ➔ [H3]
Issue airworthiness directive (AD)	-	UCA-1.5: Airworthiness directive (AD) is not issued when current LSAP is unsafe or insecure ➔ [H3]	UCA-1.6: AD is issued too late when current LSAP is unsafe or insecure ➔ [H3]
Certify airline	UCA-1.7: Certify airline which does not manage/ handle LSAP properly ➔ [H3, H4]	-	-

2) Civil Aviation Authority (CAA) UCAs

The CAA technically has responsibilities similar to that of the FAA/ EASA. In an ideal world, the FAA/ EASA and the CAA would complement each other perfectly and each would fulfill their role of overseeing the aviation industry in their respective countries. However, CAA is not as well equipped to fulfill that task as the FAA/ EASA is. It therefore makes more sense to consider the FAA/ EASA and the CAA to be asymmetrical in their responsibilities.

The CAA's responsibility is to ensure that only capable and competent airlines/ MROs are certified to perform maintenance on aircraft. It is the airline/ MRO management’s responsibility to make sure that maintenance was performed properly.

Table 2 shows the CAA unsafe control actions (UCAs). For CAA there is one UCA. UCA-2.1: Certify airline/ MRO which does not manage/ handle LSAP properly => [H3, H4]. This UCA is unsafe because it can lead to hazards H3 (*Unsafe or insecure software installed on the aircraft*) and H4 (*Software is installed improperly on the aircraft*). In turn, hazards H3 and H4 can lead to losses L1 (*Death or injury to people*) and L2 (*Damage to equipment*), as defined in section 4.2.2. Thus, if CAA certifies airline/ MRO which does not manage/ handle LSAP properly, that will lead to death or injury to people and damage to equipment.

Table 2. CAA Unsafe Control Actions

Control Action	**Providing causes hazard**	**Not providing causes hazard**	**Too soon/ too late caused hazard**
Certify airline/ MRO	UCA-2.1: Certify airline/ MRO which does not manage/ handle LSAP properly ➔ [H3, H4]	-	-

3) Manufacturer/ Software Suppliers UCAs

The manufacturer/ supplier is responsible for developing safe and secure Field Loadable Software and distributing it to airlines. It should make sure that any flaws in the software are fixed quickly and airlines are notified by means of service bulletins. Potentially unsafe control actions of the supplier include delivering unsafe/ insecure software or software updates, delivering updates too late or not at all, and issuing service bulletins too later or not at all.

Table 3 shows the manufacturer/ suppliers’ six UCAs (UCA-3.1 thru UCA-3.6). For example, let us consider UCA-3.1: Software is delivered that is unsafe or insecure => [H3, H4]. This UCA is unsafe because it can lead to hazards H3 (*Unsafe or insecure software installed on the aircraft*) and H4 (*Software is installed improperly on the aircraft*. In turn, hazards H3 and H4 can lead to losses L1 (*Death or injury to people*) and L2 (*Damage to equipment*). Thus, if software is delivered that is unsafe or insecure, that will lead to death or injury to people and damage to equipment.

Table 3. Manufacturer/ Suppliers Unsafe Control Actions

Control Action	**Providing causes hazard**	**Not providing causes hazard**	**Too soon/ too late caused hazard**
Deliver software	UCA-3.1: Software is delivered that is unsafe or insecure ➔ [H3, H4]	-	-
Deliver software update	UCA-3.2: Update is delivered that is unsafe or insecure ➔ [H3]	UCA-3.3: Update is not delivered when current version of LSAP is unsafe or insecure ➔ [H3]	UCA-3.4: Update is delivered too late when current version of LSAP is unsafe or insecure ➔ [H3]
Issue service bulletin (SB)	-	UCA-3.5: SB is not issued when current LSAP is unsafe or insecure ➔ [H3, H4]	UCA-3.6: SB is issued too late when current LSAP is unsafe or insecure ➔ [H3, H4]

4) Airline/ MRO Management UCAs

The unsafe control actions of the airline/ MRO management controller are related to its role as controller of the software management process and coordinator between various entities. The airline/ MRO management includes but not limited to Company managers, engineers, maintenance planners, information technology (IT) professionals, etc. The airline/ MRO management should ensure that it has a software management process that includes adequate protections from software tampering. The airline/ MRO management has to ensure that its aircraft are maintained in an airworthy state by assigning maintenance tasks to the airline's/ MRO's mechanics. Furthermore, it has to ensure that the tasks assigned to mechanics are carried out as required.

Table 4 shows the Airline/ MRO management unsafe control actions (UCAs). For Airline/ MRO management there are two UCAs (UCA-4.1 & UCA-4.2). For example, let us consider UCA-4.1: Software management process is not adequately protected with security mechanisms => [H3]. This UCA is unsafe because it can lead to hazards H3 (*Unsafe or insecure software installed on the aircraft*). In turn, hazard H3 can lead to losses L1 (*Death or injury to people*) and L2 (*Damage to aircraft/ equipment*). Thus, if Airline/ MRO management does not adequately protect the software management process, that will lead to death or injury to people and damage to equipment. Similar explanation works for UCA-4.2.

Table 4. Airline/ MRO Management Unsafe Control Actions

Control Action	**Providing causes hazard**	**Not providing causes hazard**	**Too soon/ too late caused hazard**
Adequately protect the software management process with security mechanisms	-	UCA-4.1: Software management process is not adequately protected with security mechanisms ➔ [H3]	-
Instruct airline/MRO mechanics to perform LSAP update	-	UCA-4.2: Instruction to update LSAP is not provided when AD or SB requires it, and airline/ MRO mechanics are not tasked with it ➔ [H3]	-

5) Airline/ MRO Mechanics UCAs

The mechanic's responsibility is to keep the aircraft in an airworthy condition. With respect to field loadable software this requires making sure that the software is installed according to the type certificate and that it is up to date with service bulletins.

Table 5 shows the Airline/ MRO mechanics' unsafe control actions (UCA-5.1 thru UCA-5.4). Let us consider UCA-5.1: LSAP not installed as specified by type certificate => [H3, H4]. This UCA is unsafe because it can lead to hazards H3 (*Unsafe or insecure software installed on the aircraft*) and H4 (*Software is installed improperly on the aircraft*). In turn, hazards H3 and H4 can lead to losses L1 (*Death or injury to people*) and L2 (*Damage to equipment*). Thus, if Airline/ MRO mechanics do not install LSAP as specified by type certificate, that will lead to death or injury to people and damage to aircraft/equipment. Similar explanation works for UCA-5.2 thru UCA-5.4.

Table 5. Airline/ MRO Mechanics Unsafe Control Actions

Control Action	**Providing causes hazard**	**Not providing causes hazard**	**Too soon/ too late caused hazard**
Initial LSAP install	UCA-5.1: Install LSAP not as specified by type certificate ➔ [H3, H4]	UCA-5.2: Software install is not performed when component is new ➔ [H4]	-
LSAP update	UCA-5.3: Update LSAP not as specified by AD or SB ➔ [H3, H4]	UCA-5.4: Software update not provided in accordance with AD or SB ➔ [H3]	-

6) Pilots UCAs

Pilots have many responsibilities, but two of those are relevant to our analysis which are performing pre-flight check and operating the aircraft properly. For example, verifying that the EGPWS (Enhanced Ground Proximity Warning System) is in working order before taking off and avoiding collisions with terrain, obstacles, and other aircraft. The potentially unsafe control actions are not performing the pre-flight check and not properly reacting to an alert.

Table 6 shows the pilots' unsafe control actions (UCA-6.1 thru UCA-6.4). Let us consider UCA-6.1: Pre-flight check not performed when LSAP does not conform to type certificate => [H3, H4]. This UCA is unsafe because it can lead to hazards H3 (*Unsafe or insecure software installed on the aircraft*) and H4 (*Software is installed improperly on the aircraft*). In turn, hazards H3 and H4 can lead to losses L1 (*Death or injury to people*) and L2 (*Damage to equipment*). Thus, if pilots do not perform pre-flight check when LSAP does not conform to type certificate, that will lead all to death or injury to people and damage to aircraft/equipment. Similar explanation works for UCA-6.2 thru UCA-6.4.

Table 6. Pilots Unsafe Control Actions

Control Action	**Providing causes hazard**	**Not providing causes hazard**	**Too soon/ too late caused hazard**
Perform pre-flight check of LSAP	-	UCA-6.1: Pre-flight check not performed when LSAP does not conform to type certificate ➔ [H3, H4]	-
Properly react to an alert	UCA-6.2: React to alert when there is incorrect or conflicting alert ➔ [H1]	UCA-6.3: Not properly reacted to alert when there is imminent unusual and potentially dangerous circumstance ➔ [H1]	UCA-6.4: Reacted too late to alert when there is imminent unusual and potentially dangerous circumstance ➔ [H1]

7) LSAP (Software) UCAs

The unsafe control actions of the LSAP controller are related to displaying software configuration and issuing alert. For example, issuing a terrain alert by EGPWS. The LSAP displays the loaded software configuration (version) to the mechanics and issues (notifies) alerts to the pilots of any unusual and potentially dangerous circumstances of the aircraft.

Table 7 shows the pilots' unsafe control actions (UCA-7.1 thru UCA-7.3). Let us consider UCA-7.1: LSAP (Software) does not provide alert when unusual and potentially dangerous circumstance is imminent => [H1]. This UCA is unsafe because it can lead to hazards H1 (*Loss of control of aircraft*). In turn, hazards H1 can lead to losses L1 (*Death or injury to people*) and L2 (*Damage to equipment*). Thus, if LSAP does not provide alert when unusual and potentially dangerous circumstance is imminent, that will lead to death or injury to people and damage to aircraft/equipment. Similar explanation works for UCA-7.2 and UCA-7.3.

Table 7. LSAP (Software) Unsafe Control Actions

Control Action	**Providing causes hazard**	**Not providing causes hazard**	**Too soon/ too late caused hazard**
Issue alerts of unusual and potentially dangerous circumstances	-	UCA-7.1: Does not provide alert when unusual and potentially dangerous circumstance is imminent ➔ [H1]	UCA-7.2: Warn too late of imminent unusual and potentially dangerous circumstance ➔ [H1]
Display the loaded software version (configuration)	-	UCA-7.3: Does not display the loaded software version (configuration) ➔ [H4]	-

4.2.5. Loss Scenarios for Unsafe Control Actions

Once unsafe control actions have been identified, the next step is to identify loss scenarios. A loss scenario describes the causal factors that can lead to the unsafe actions and to hazards. Two types of loss scenarios must be considered: (a) why would unsafe control actions occur? and (b) why would control actions be improperly executed or not executed, leading to hazards?

The software is considered to be unsafe or insecure if it can lead to one of the systems hazards, which are identified in section 4.2.2. The software can be unsafe in different ways: it may cause the component or line replaceable unit (LRU) on which it is installed to stop working properly, and/or other components to which it is connected to stop working.

In this section, the loss scenarios have been identified for each unsafe control actions. The complete list of the loss scenarios analysis is attached as Appendix 1 of this paper. These identified loss scenarios (hazards) can be prevented by enforcing the system-level constraints, which are indicated in section 4.2.3. A system-level constraint specifies system conditions or behaviors that need to be satisfied to prevent hazards/ losses. However, in line with the scope of this study, further analysis will be carried out for those related to the security of the LSAP distribution system in the next section.

In STPA-Sec analysis, the control structure usually begins at a very abstract level and is iteratively refined to capture more detail about the system. The individual scenarios could naturally be refined to more detail level if desired and we decide to stop at the point where enough detail is available to devise reasonable security controls. Thus, we need to further refine the LSAP (software) controller in order to capture more detail and to see its interactions with the mechanic. Here, EGPWS (Enhanced Ground Proximity Warning System) could be taken as example of LSAP controller.

Figure 6 shows a detailed view of LSAP (Software) controller and its interactions with the mechanic. The mechanic installs/updates an Operational Program software onto the line replaceable unit (LRU)/component using the Software Logger. For some LSAPs the mechanic can verify the version of installed software from the software loader/component. The mechanic can also update the databases onto the LRU using the Database Manager. Some of the databases, such as terrain database (TDB) and navigation database (NDB) are updated regularly in fixed interval. The LSAP controller gets the aircraft data input from the aircraft/avionics systems for its operations/calculations. Then it sends alerts, database information, software version, device status, etc. to the Multi-Functional Display (MFD) unit. Here the mechanic can read the alerts, software versions or database information after installing/updating the LSAP. Currently, there is no way of determining the exact LSAP version installed on the device without relying on the installed software itself. This is considered as a clear security vulnerability as an attacker can send a modified software to be installed using the same software version number as the original software.

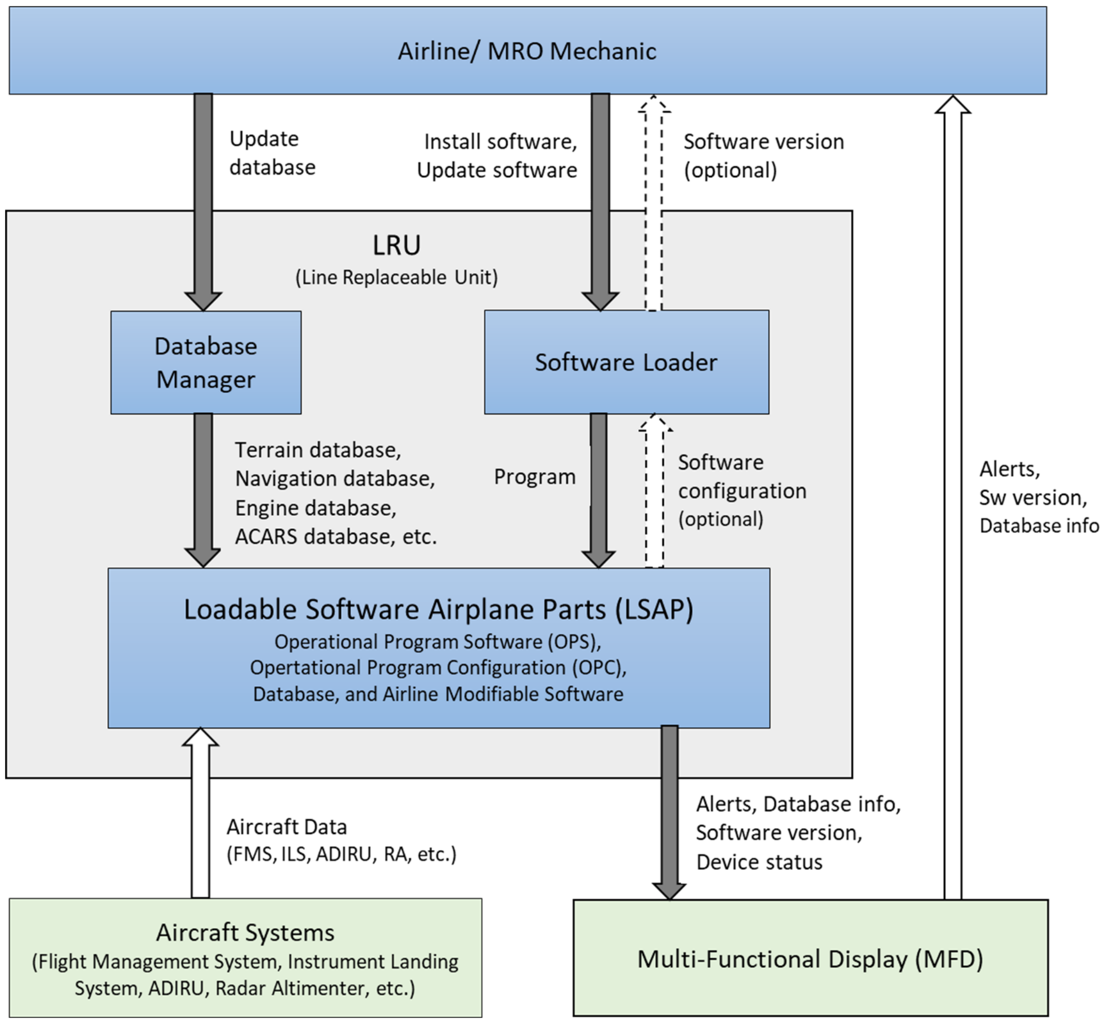
Airline/ MRO Mechanic
Update database
Install software, Update software
Software version (optional)
LRU
(Line Replaceable Unit)
Database Manager
Software Loader
Terrain database, Navigation database, Engine database, ACARS database, etc.
Program
Software configuration (optional)
Alerts, Sw version, Database info
Loadable Software Airplane Parts (LSAP)
Operational Program Software (OPS), Opertational Program Configuration (OPC), Database, and Airline Modifiable Software
Aircraft Data (FMS, ILS, ADIRU, RA, etc.)
Alerts, Database info, Software version, Device status
Aircraft Systems (Flight Management System, Instrument Landing System, ADIRU, Radar Altimenter, etc.)
Multi-Functional Display (MFD)

Figure 6. Detailed view of the LSAP and its interactions with the mechanic

(This figure refines the LSAP control structure further for STPA-Sec analysis)

4.2.6. Security Vulnerabilities in the LSAP Distribution System

In line with the scope of this research, this section presents a selection of loss scenarios or security vulnerabilities that are specific to the security of the LSAP distribution system. That is, we focus on the loss scenarios in which the analysis suggests that an attacker could cause one of the system losses through the manipulation of an LSAP component(s). The attacker could be anyone with the necessary expertise: a dishonest employee, a spy, a terrorist, a criminal, a security researcher, etc.

Scenario 1:

Once the software is received from the manufacturer/supplier, the airline uses the embedded digital signatures to verify that the source of the software and it has not been modified. Then the airline signatures are applied for the latest design airplanes that adapted technological advances. However, for previous design airplanes that do not have technology for digital signatures, there are currently no security mechanisms in place to verify the source of the software and to prevent the tampered with software from installing on the aircraft. Also, once the software is installed, there is no way to verify whether the software has been tampered with or not. An attacker could take advantage of this and either tamper the software while stored or send a tampered with software to maintenance laptops during transmission.

Scenario 2:

After the software is received from the manufacturer/supplier and saved or stored in the company servers, the engineers get access to the data distribution server for the receiving inspection and to the LSAP storage server to for managing the LSAP configuration and effectivity. To do this they use assigned username and password. However, using passwords alone cannot be considered secure [40] [42]. An attacker could carry out spoofing attack and get access to the servers to tamper the LSAP by acting as the airline engineers.

Scenario 3:

To download/transfer the software from the LSAP storage server to the maintenance laptops, the mechanics use the assigned username and password to get access to the maintenance laptops as well as the LSAP storage server. Passwords alone are not a secure [40] [42]. An attacker could carry out man-in-the middle attack and send a tampered LSAP to the mechanics or modify the software when it is in the maintenance laptop.

Therefore, the STPA-Sec analysis showed that there are two critical security vulnerabilities in the aircraft software distribution system:

1) There is no security mechanism for the previous design airplanes to authenticate the identity of the sender of the software and to ensure that the original content of the document is unchanged.

2) Password-based single-factor authentication is used for accessing the ground-based software servers (i.e., data distribution management and LSAP storage servers) as well as maintenance laptops.

The proposed security mechanism to enhance the existing security mechanism for the LSAP software distribution as well as the proposed artifact, i.e., security architecture, will be described and discussed in the next chapter.

CHAPTER FIVE - PROPOSED MULTI-LAYER SECURITY MECHANISM

5.1. Proposed Solutions

The STPA-Sec analysis showed that there are two critical security vulnerabilities in the aircraft software distribution system that should be addressed: (1) Currently there is no security mechanism for the previous design airplanes to authenticate the identity of the sender of the software and to ensure that the original content of the document is unchanged. (2) Password-based single-factor authentication is used for accessing the LSAP servers as well as maintenance laptops.

To address the first security vulnerability, we apply multi-layer security mechanisms. Multi-layer security is the concept of protecting a computer network with a series of defensive mechanisms such that if one mechanism fails, another will already be in place to thwart an attack. It is a defense mechanism that mitigates, delays, or prevents threats. The cybercriminals often target the end-users and entry points. So, the best security approach would be to cover each layer so that the confidential information remains safe. By utilizing layered defense strategy, any attack can be prevented or at least detected [32] [33] [34].

To address the second security vulnerability, we apply multi-factor authentication mechanism (MFA). MFA is a strong authentication method that requires two or more factors to gain access to the system. Each factor must come from a different category of the authentication methods (i.e., something you know, something you have, and something you are). Therefore, MFA is applied to airline distribution system as part of the multi-layer security mechanisms.

5.2. Multi-layer Security Mechanism

Multi-layer security, also known as multi-level security or defense in depth, refers to the system that uses numerous components to shield the IT infrastructure. Though the multi-layered security and defense in depth terms are generally used together, they refer to a security plan designed to mitigate or prevent breaches in data security threats. Multi-level security is based on the idea of using numerous security solutions to guard the system from threats. However, a deep defensive security policy should be aimed at preventing threats from becoming dangerous.

The multiple layers of network security ensure that the defense component protects the data in the event of any failure or a loophole in the system. In a multi-layered security strategy, every single layer in the multi-level security approach focuses on key areas of security that provide a better chance to prevent cybercrimes. The individual layers in the multi-layered security strategy focus on areas that are vulnerable to cyberattacks [33]. .

In the cybersecurity paradigm, Defense in Depth correlates to detective and protective measures designed to impede the progress of a cyber intruder while enabling an organization to detect and respond to the intrusion with the goal of reducing and mitigating the consequences of a breach. Defense in Depth employs a holistic approach to protect all assets, while taking into consideration its interconnections and dependencies, and using an organization's available resources to provide effective layers of monitoring and protection based on the business's exposure to cybersecurity risks [35].

A threat-actor, through intent, capability, and/or opportunity, poses a threat to a system by compromising an organization's systems through its operations, personnel, and/or technology and exploiting an existing weakness or vulnerability. Security countermeasures, based on best practices and standards, protect critical assets through multiple layers of defense - thereby improving protection for operations, personnel, and technology [35].

We apply the multi-layer security mechanisms and controls at the network, system, application, and physical layers to provide information assurance. These include policy and security management, application security, data security, platform security, network and perimeter security, physical security, and user security [38].

The multi-layer security mechanism is built by applying the proposed security mechanisms and controls to the existing security resources and controls. The following are multi-layer security controls (security countermeasures) that are applied for LSAP distribution system, which are adapted from the Open Enterprise Security Architecture (O-ESA) Framework [38]:

(1) Border Protection

The existing system has firewalls and VPN for the border protection. Firewalls are used to isolate the internal network from the internet, whereas VPN encrypts data as it traverses un-trusted networks.

(2) Access Control

Access to the maintenance laptops as well as the company DDM (data distribution management) server and LSAP storage server are controlled based on the user's identity (authentication) and user's entitlements/ privileges (authorization).

a) Authentication

The existing system uses username and password for accessing the servers as well as the maintenance laptops.

We apply the multi-factor authentication (MFA) into the new security architecture on top of the username and password, to provide adequate protection to the LSAP distribution system.

b) Authorization

Only authorized users are permitted to gain access to do entitled functions on the DDM server, LSAP storage server and maintenance laptops. The privileges will be non-administrative.

(3) Detection (Activity Monitoring)

The existing system uses security logs to monitor the aircraft web server or onboard network server (ONS) for suspicious activities for latest design aircraft types that adapted technology advancements.

The Intrusion Detection & Prevention System (IDPS) is used to deeply monitor the packets on the network, and when any malicious activity is detected, it will log and block it. It will also report this activity to alert the system administrators. Also, IDPS forms the second defense line in securing the core network.

(4) Content Control

The existing system has Anti-Virus/Anti-Malware software installed on the maintenance laptops as well as on the DDM server and LSAP storage server to identify, block and remove viruses embedded in files. USB drives will be automatically scanned by anti-malware controls when plugged into the maintenance laptop.

(5) Auditing

In the existing system, the security logs are downloaded and analyzed regularly in support of security investigations and risk assessments, for the latest design airplanes. Additionally, LSAP storage server is regularly audited for the content and effectivity of software parts.

(6) Cryptography

The existing system uses digital signature to authenticate the identity of the sender/signer of the software and to ensure that the original content of the document is unchanged. Digital signatures are used to transfer software from manufacturer (manufacturer DDM server) to the airline (company DDM server). Airline approval digital signature is applied to protect the software parts within the airline repository and during software transfer to the airplane.

For the latest design airplanes, the digital signatures are signed by the airline at the DDM workstation to be distributed via maintenance laptop and verified onboard the airplane before installation to the LRU.

For the previous design airplanes, currently there is no technological capability for digital signatures or no other approved security mechanism to authenticate the identity of the sender/signer of the software and to ensure that the original content of the document is unchanged. Hence, the multi-layer security mechanism will be used to prevent or detect any attack. For the future, a secure hash function (i.e., keyed hash or message authentication code “MAC”) could be used for the authentication and verification of software for the previous design airplanes, but it needs further research and needs to get approval from the regulatory agencies.

5.3. Security Architecture Design

The main purpose of security architecture is integration between different security elements (i.e., network, information, etc.) and providing a single document that specifies the security services [37]. Security Architecture is a cohesive security design, which addresses the requirements (e.g., authentication, authorization, etc.) and in particular the risks of a particular environment or scenario and specifies what security controls are to be applied where.

The security architecture includes the specific controls and their strategic placement within the network or systems to establish layers of security - Defense in Depth. Network diagrams and information flow diagrams that include all systems and their interconnections couple with the physical inventory to provide an operational-level understanding of the information flows within the network. Then overlay this with the protection levels for each system or subsystem that was assigned during inventory activities to help determine the controls to put in place to protect the system without compromising or degrading its performance [35].

The security architecture is designed by applying the multi-layer security mechanisms. It is created using network diagrams. Figure 7 shows the proposed security architecture for the LSAP distribution system in the airlines. Software parts are received electronically from the manufacturer/ supplier data distribution management (DDM) server to the company DDM server. Upon receipt the responsible company engineers access the DDM server from their desktop and inspect the contents and validate the digital signature Once the software is verified and approved, the engineers save/store the software in the LSAP storage server and manage the configuration and effectivity of the software. When the software is scheduled to be installed on the aircraft, the mechanics access and download the software from the LSAP storage server to the maintenance laptops. Then the software parts are installed into the LRUs on the aircraft. The security controls are applied using the multi-layer security mechanisms (refer to section 5.2) and as indicated on the below security architecture.

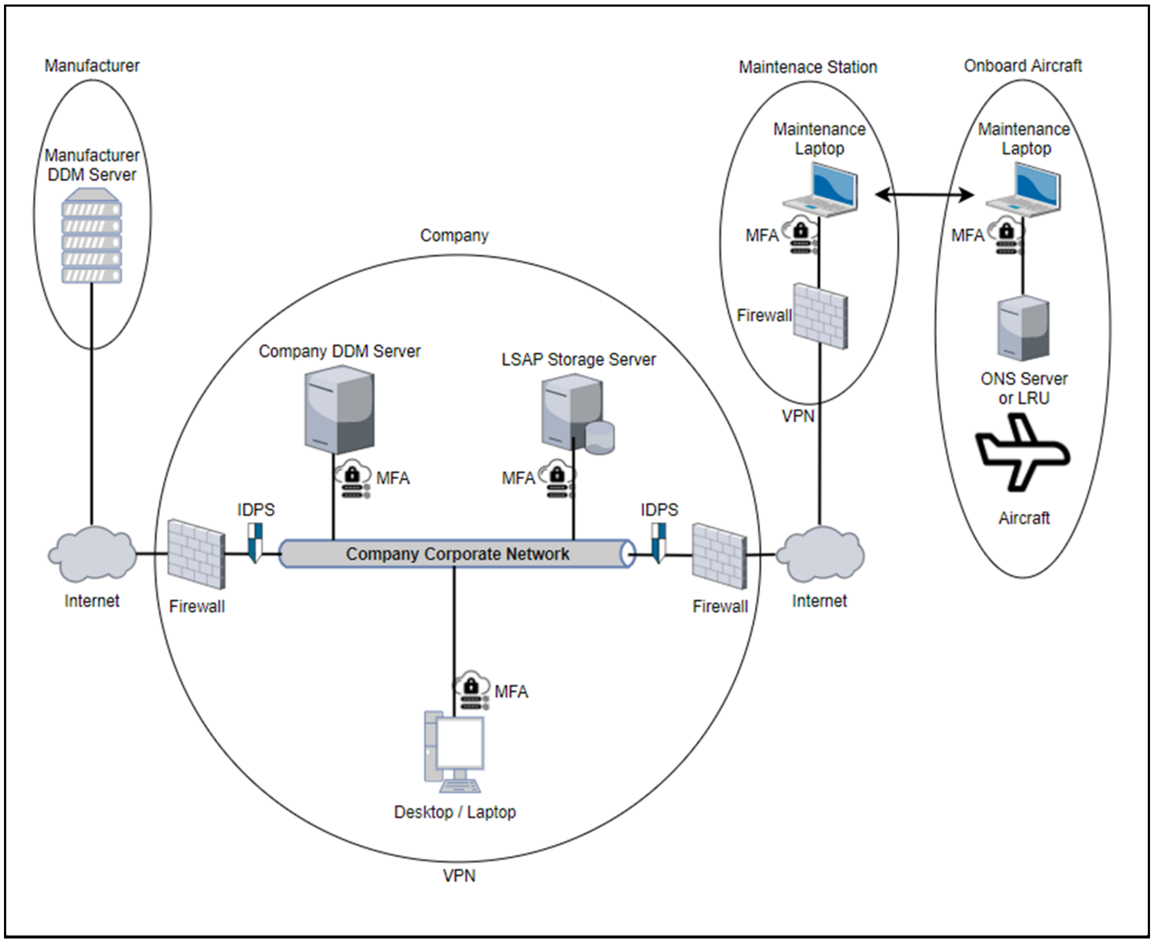

Figure 7. Security Architecture for LSAP Distribution System

Source: Own - created using "Draw.io" app

CHAPTER SIX -
EVALUATION OF THE PROPOSED MULTI-LAYER SECURITY MECHANISM

This chapter evaluates how well the artifact (i.e., the proposed security architecture) supports a solution to the problem. This study employs the Cyber Security Evaluation Tool (CSET). CSET is a product that assists organizations in protecting their key cyber assets.

6.1. Cyber Security Evaluation Tool (CSET)

CSET is a software tool for performing cybersecurity evaluations of an enterprise and industrial control cyber systems. It is easy to install and use on a stand-alone laptop or workstation. It was designed to help asset owners identify vulnerabilities and improve the organization's overall cybersecurity posture by guiding them through a series of questions that represent network security requirements and best practices. The presented requirement questionnaires are based on selected industry standards, common requirements, and the network diagram (or network topology and architecture) [39].

It incorporates a variety of available standards from organizations, such as NIST, Department of Defense (DoD), etc. When the tool user selects one or more of the standards, CSET opens a set of questions to be answered. The answers to these questions are compared against a selected security assurance level, and a detailed report is generated that shows areas for potential cybersecurity improvement.

The tool derives the recommendations from a database of cybersecurity standards, guidelines, and practices. The Analysis dashboard provides interaction with graphs and tables that present the assessment results in both summary and detailed form. It also provides the top areas of concern that are prioritized based on current threat information.

6.2. Evaluation Results and Implications

The evaluation was carried out for both pre and post proposed solutions scenarios. It is the simulation of the security architecture based on the following criteria:

(a) Cybersecurity Standards: Standard.

(b) Security Assurance Level: High (i.e., Confidentiality = High, Integrity = High, and Availability = High)

(c) Diagram: Security Architecture

(d) Questions: The system generated 83 requirements and 291 questions based on the selected criteria. The questions were answered and evaluated for the pre-proposed solution and post-proposed solution security architectures individually.

(e) Analysis and Reports:

The overall score is indicated using the percentage, which indicates the overall compliance of the system or architecture with the security standards and is calculated as follows:

$$Overall\ Score\ (\%) = \frac{number\ of\ passed\ (\text{"Yes" or "ALT" answered})\ questions}{total\ number\ of\ questions}$$

The evaluation results for pre and post proposed solutions show the overall score of 86% and 96% respectively. This indicates that the compliance of the system with the security standards of the LSAP distribution system has improved by 10% when the proposed solutions are applied. This implies the efficacy and efficiency of the proposed multi-layer security mechanism and hence, the security architecture created by applying the multi-layer security mechanism.

Figure 8 shows the result of the CSET evaluation by categories of the security controls before the proposed solutions. Whereas Figure 9 shows the result of the CSET evaluation by categories of the security controls after the proposed solutions.

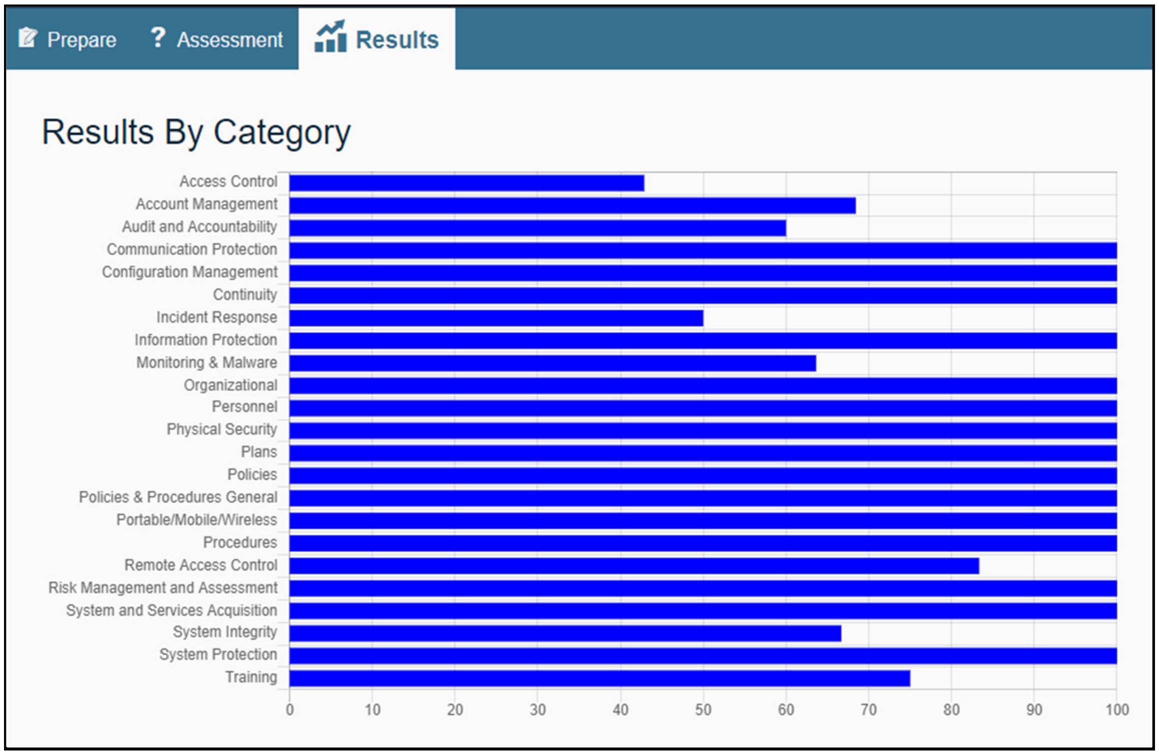

Figure 8. Evaluation Results Before the Proposed Solutions

Based on the CSET assessments

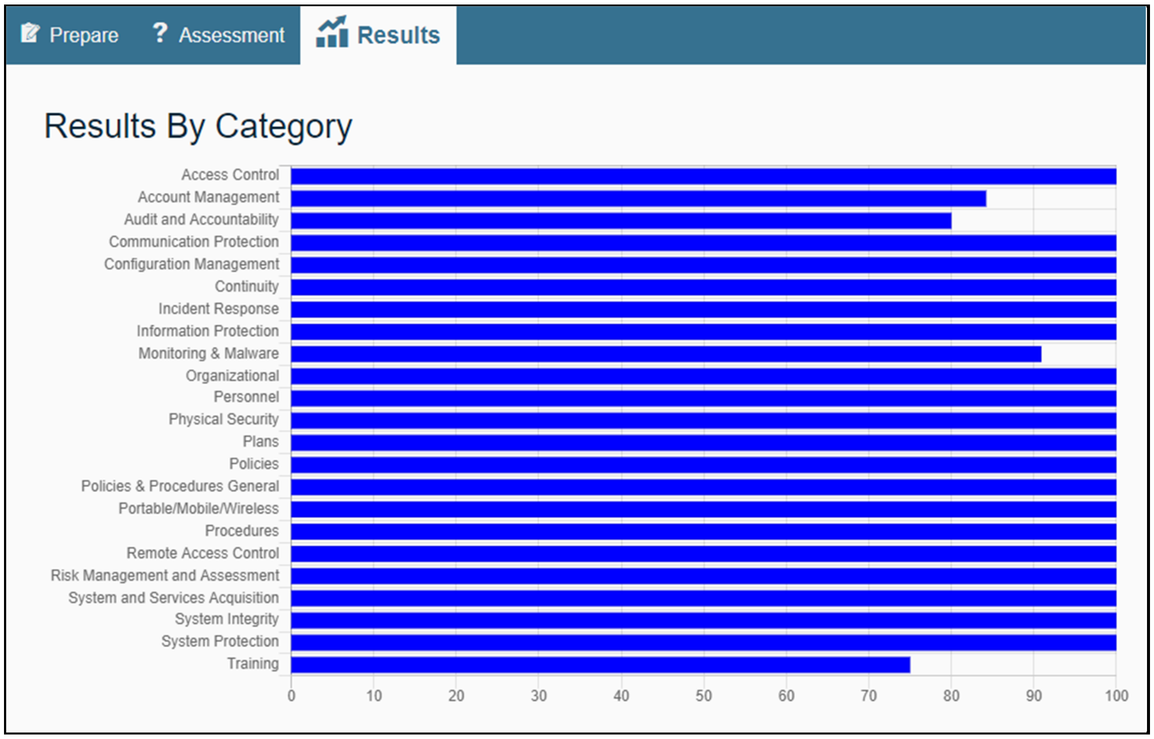

Figure 9. Evaluation Results After the Proposed Solutions

Based on the CSET assessments

CHAPTER SEVEN - CONCLUSIONS AND RECOMMENDATIONS

This is the final chapter of this study, and it presents the conclusion and recommendations of the research, which was carried out on the Multi-Layer Security Mechanisms for Commercial Aircraft Software Distribution System.

7.1. Conclusions

This thesis reviewed the aircraft software distribution system and its inherent security in commercial aviation industry, using airlines a case study. In this study, the comprehensive security analysis of the aircraft software distribution system was carried out using the systems approach called Systems Theoretic Process Analysis (STPA) in order to identify potential security vulnerabilities and their impact. The review covered the loadable software airplane parts (LSAP) and the related processes and procedures. A high-level description of the LSAP life cycle and an insightful hierarchical control structures were modeled and used to determine the potentially unsafe control actions. The loss scenarios were identified for each unsafe control actions.

This study found out two critical security vulnerabilities in the aircraft software distribution system. These are: (1) currently, there are no security mechanisms in place to verify the source of the software for previous design airplanes and (2) password-based single-factor authentication is used for accessing the ground-based software servers as well as maintenance laptops. In addition to these, the analysis identified the loss scenarios for the unsafe control actions of the system controllers (i.e., FAA/EASA, CAA, manufacturer/supplier, airline/MRO management, airline/MRO mechanics, and pilots). This study also showed that these loss scenarios can be prevented by enforcing the system-level constraints.

To address the critical security issues or vulnerabilities, multi-layer security approach was proposed and applied. Multi-level security approach is based on the idea of using numerous security solutions to guard the system from threats. The security architecture was designed using the proposed multi-level security mechanisms.

The proposed multi-layer security mechanism and the security architecture were evaluated using the Cyber Security Evaluation Tool (CSET) that provides systematic approach to evaluating the security posture of the cyber systems and networks. The evaluation result showed that the proposed solutions are highly effective and efficient.

The findings of this study will redound to the commercial aviation in general and benefit the airline management as well as the security personnel in airlines in particular. The major contribution of this study is applying a multi-layer security mechanism for the aircraft software distribution system, which enhances the existing security mechanisms and provides adequate security protection.

7.2. Recommendations

This study demonstrated that the multi-layer security approach is highly effective in solving the security issues or vulnerabilities of the aircraft software distribution system. However, currently there is no approved security mechanism for the previous design airplanes to authenticate the identity of the sender of the software and to ensure that the original content of the document is unchanged. We recommend that a cryptographic hash function (i.e., keyed hash or message authentication code "MAC") could be used for the authentication and verification of software for the previous design airplanes. At the same time, we recommend that it needs further research as it should get the regulatory agencies' approvals to be applied for this purpose.

Due to the time constraint, this study was based mainly on the common data and used airlines as a case study. It did not study a specific airline's software management process and practices in detail. Such study is necessary in the future to determine how serious the loss scenarios and problems really are that this analysis found.

The analysis presented in this research did not study the software development and implementation. It focused on the security of the aircraft software distribution system. It would be an interesting topic for further research to see how the STPA-Sec can be applied to software directly. It is also interesting to identify the vulnerabilities in the aircraft software development processes and in the software itself.

References

[1] Willis Towers Watson, “Advancing Cyber Resilience in Aviation: An Industry Analysis”, 2020

[2] US Cybersecurity and Infrastructure Security Agency (CISA), “Critical Infrastructure Sectors”, https://www.cisa.gov/critical-infrastructure-sectors (link as of October 21, 2020)

[3] IATA, “Importance of Air Transport to Ethiopia”, https://www.iata.org/en/pressroom/pr/2020-03-04-02/ (link as of March 04, 2020)

[4] R. K. Rajasekaran and E. Frew, “Cyber security challenges for networked aircraft”, in Proceedings of the Integrated Communications, Navigation and Surveillance Conference (ICNS). IEEE, 2017, pp. 1–15.

[5] H. Duchamp, I. Bayram, and R. Korhani, “Cyber-security, a new challenge for the aviation and automotive industries”, 2016.

[6] US Government Accountability Office (GAO), “Aviation Cybersecurity: FAA Should Fully Implement Key Practices to Strengthen Its Oversight of Avionics Risks”, 2020.

[7] Anna Baron Garcia, “Building and Integrating an Information Security Trustworthiness Framework for Aviation Systems’, 2019.

[8] Tewodros Getaneh, “Cyber Security Practices and Challenges at Selected Critical Infrastructures in Ethiopia: Towards Tailoring Cyber Security Framework”, 2018.

[9] Scott Lintelman, Richard Robinson, Mingyan Li, and Krishna Sampigethaya, “Formal Methods for Trustworthy Skies: Building Confidence in the Security of Aircraft Assets Distribution”, 2008.

[10] Richard Robinson, Mingyan Li, Scott Lintelman, Krishna Sampigethaya, Radha Poovendran, David von Oheimb, Jens-Uwe Bußer, and Jorge Cuellar, “Electronic Distribution of Airplane Software and the Impact of Information Security on Airplane Safety”, 2007.

[11] FAA, “Software Management During Aircraft Maintenance”, Advisory Circular AC 43-216, 2017.

[12] RTCA / EUROCAE, “Software Considerations in Airborne Systems and Equipment Certification”, DO- 178C/ED-12C, 2011.

[13] Peter Skaves, “Information for Cyber Security Issues Related to Aircraft Systems”, 2013.

[14] David von Oheimb, Monika Maidl, and Richard Robinson, “Security Architecture and Formal Analysis of an Airplane Software Distribution System, 2008.

[15] Jonas Helfer, “A Systems Approach to Software Security in Aviation”, 2016.

[16] J. W. Creswell, V. L. P. Clark, “Designing and Conducting Mixed Methods Research”, 2007.

[17] Mark Dowd, John McDonald and Justin Schuh, “The Art of Software Security Assessment: Identifying and Preventing Software Vulnerabilities”, 2006, Pages 22-24.

[18] FAA, “Software Approval Guidelines”, Order 8110.49A, 2018.

[19] RTCA/ EUROCAE, “Information Security Guidance for Continuing Airworthiness”, DO-355, 2014.

[20] RTCA/ EUROCAE, “Airworthiness Security Process Specification”, DO-326A, 2010.

[21] FAA, “Airborne Software Development Assurance Using EUROCAE ED-12( ) and RTCA DO-178( )”, Advisory Circular AC 20-115D, 2017.

[22] SAE, “Guidance for Security of Loadable Software Parts Using Digital Signatures”, ARINC Report 835-1, 2014.

[23] Boeing Commercial Airplanes, “Statistical Summary of Jet Airplane Accidents, 1959-2014”, 2015.

[24] Andrew Kornecki, and Mingye Liu, “Fault tree analysis for safety/security verification in aviation software”, 2013.

[25] ISO/IEC, “Common Criteria for Information Technology Security Evaluation”, Version 3.1, Revision 5, 2017.

[26] Ken Peffers, Tuure Tuunanen, Marcus Rothenberger, Samir Chatterjee, “A Design Science Research Methodology for Information Systems Research”, 2008.

[27] Alan Hevner, Salvatore March, Jinsoo Park, Sudha Ram, “Design Science in Information Systems Research”, 2004.

[28] FAA, “Airworthiness and Operational Authorization of Aircraft Network Security Program (ANSP)”, Advisory Circular AC 119-1, 2015.

[29] Open Security Architecture (OSA), “IT Security Architecture”, https://www.opensecurityarchitecture.org/cms/definitions/it-security-architecture, (link as of June 2021).

[30] Nancy Leveson, “A New Accident Model for Engineering Safer Systems”, Safety Science 42.4, 2004, pages 237-270.

[31] Nancy Leveson and John Thomas, “STPA Handbook”, 2018.

[32] Ahmed S. Mustafa, Kalaivani Chellappan, Mohammed J. Mohammed, Aqeel M. Thajeel, “Layered Defense Approach: Towards Total Network Security”, 2015.

[33] Cyber Chasse, “ How A Multi-Layered Security Strategy Can Protect Your Business”, https://cyberchasse.com/how-a-multi-layered-security-strategy-can-protect-your-business/ (link as of June 2021).

[34] Blue Ridge Technology, “Multi-Layered Security Approach”, https://www.blueridge.tech/2019/03/06/multi-layered-security-approach/ (link as of June 2021).

[35] Homeland Security, “Recommended Practice: Improving Industrial Control System Cybersecurity with Defense-in-Depth Strategies”, 2016.

[36] National Institute of Standards and Technology (NIST), “Framework for Improving Critical Infrastructure Cybersecurity”, Version 1.1, 2018.

[37] Shahram Jalaliniya and Farzaneh Fakhredin, “Enterprise Architecture and Security Architecture Development”, 2011.

[38] The Open Group, “Open Enterprise Security Architecture (O-ESA) – A Framework and Template for Policy-Driven Security”, Van Haren Publishing (VHP), 2011.

[39] Cybersecurity & Infrastructure Security Agency (CISA), “Assessments: Cyber Security Evaluation Tool (CSET)”, https://us-cert.cisa.gov/ics/Assessments (link as of June 2021)

[40] Charlie Jacomme and Steve Kremer, “An Extensive Formal Analysis of Multi-Factor Authentication Protocols”, 2018.

[41] Boeing, “Onboard Loadable Software”, https://www.boeing.com/commercial/aeromagazine/aero_05/textonly/ps02txt.html (link as of June 2021)

[42] Civil Aviation Safety Authority, “Administration of Aircraft & Related Ground Support Network Security Programs”, Civil Aviation Advisory Publication CAAP 232A-1, 2013.

[43] IATA, “Best Practices for Loadable Software Management and Configuration Control”, 2013.

Appendix 1 -

Lists of Loss Scenarios for Unsafe Control Actions

1) <u>Loss Scenarios for FAA/EASA UCAs</u>

<u>**UCA-1.1, 1.2:** FAA/EASA provides a type certificate for unsafe software or update.</u>

(a) Due to incorrect feedback or incorrect instructions from another controller:

- ➢ A type certificate is provided for unsafe software because the software tested on aircraft is not the exact software certified/delivered. This could happen if the software is not tracked appropriately with versioning.

(b) Due to process model flaw or algorithm flaw:

- ➢ A type certificate is provided for unsafe software because the designated engineering representative (DER) feels pressure to sign-off on certificate because of job insecurity.
- ➢ A type certificate is provided for unsafe software because tests/checks are not thorough enough because the supplier has good track record with FAA/EASA.
- ➢ A type certificate is provided for unsafe software because tests and inspections are insufficient to discover a hazardous flaw.

(c) Control action is issued but not effective:

- ➢ Not Applicable

<u>**UCA-1.3, 1.4:** FAA/EASA provides certification too late or not at all.</u>

(a) Due to incorrect feedback or incorrect instructions from another controller:

- ➢ FAA/EASA certifies a critical update too late or not at all because the request for certification gets lost in the mail (or someone's inbox).
- ➢ FAA/EASA certifies a critical update too late or not at all because the supplier did not submit all required documentation.

(b) Due to process model flaw or algorithm flaw:

- FAA/EASA certifies a critical update too late or not at all because it does not have enough inspectors to process all the certification requests in time.
- FAA/EASA certifies a critical update too late or not at all because the application gets stuck on someone's desk at the FAA/EASA and nobody else takes over the task.
- FAA certifies a critical update too late or not at all because the application is not treated with the necessary urgency.

(c) Control action is issued but not effective:

- The certification is issued but it gets lost in the mail (or someone's inbox) and nobody notices for several days or weeks.

UCA-1.5, 1.6: FAA/EASA issues airworthiness directive too late or not at all.

(a) Due to incorrect feedback or incorrect instructions from another controller:

- The airworthiness directive (AD) is issued too late or not at all because no incidents are reported to the FAA/EASA despite hazard.
- The AD is issued too late or not at all because reported incidents never make it to the FAA due to a flaw in the system, due to no follow-up for the incidents.
- The AD is issued too late or not at all because reported incidents do not contain all necessary information to pinpoint the problem source.

(b) Due to process model flaw or algorithm flaw:

- The AD is issued too late or not at all because reported incidents do not get dealt with in efficient manner.
- The AD is issued too late or not at all because discovered hazards are not considered serious enough to warrant an AD.

(c) Control action is issued but not effective:

- FAA/EASA issues an AD, but it is not effective because it was not noticed by all owners/pilots of the concerned aircraft.

2) **Loss Scenarios for CAA UCAs**

- Discussed in Chapter 4.

3) **Loss Scenarios for Manufacturer/Supplier UCAs**

UCA-3.1: Manufacturer/Supplier delivers unsafe software.

(a) Due to incorrect feedback or incorrect instructions from another controller:

- The manufacturer/supplier delivers unsafe/insecure software because the FAA/ EASA certified without extensive tests.

(b) Due to process model flaw or algorithm flaw:

- The manufacturer/supplier delivers unsafe/insecure software because an insider/attacker inserted malicious code into the code base.
- The manufacturer/supplier delivers unsafe/insecure software because the wrong version of the software was published by accident.
- The manufacturer/supplier delivers unsafe/insecure software because some testes were skipped, or the testing setup did not work as intended.

(c) Control action is issued but not effective:

- The manufacturer/supplier publishes safe and secure software, but the airline/MRO mechanic installed the wrong version/software by accident.

UCA-3.2: Manufacturer/Supplier delivers update when the update is unsafe/insecure.

(a) Due to incorrect feedback or incorrect instructions from another controller:

- Not applicable.

(b) Due to process model flaw or algorithm flaw:

- The manufacturer/supplier ships an unsafe or insecure update because the bug fix/update introduced new hazards.
- The manufacturer/supplier ships an unsafe or insecure update because the update did not fix the underlying problem but only patched one instance of bug.

(c) Control action is issued but not effective:

- The manufacturer/supplier issued a safe and secure update, but an attackers tricked the airline into downloading a modified version from their own website.

UCA-3.3, 3.4: Manufacturer/Supplier ships update too late or not at all.

(a) Due to incorrect feedback or incorrect instructions from another controller:

- The manufacturer/supplier ships an update too late or not at all because the supplier is not aware of hazards in original software.
- The manufacturer/supplier decides not to ship an update for a known issue until the FAA issues airworthiness directive.

(b) Due to process model flaw or algorithm flaw:

- The manufacturer/supplier ships an update too late because the update depends on an engineer that is unreachable or because of a lack of experienced engineer.
- The manufacturer/supplier does not ship an update because it is too costly, and the company is in financial trouble.

(c) Control action is issued but not effective:

- The manufacturer/supplier issues an update, but the update is not installed by the airline/MRO because a corresponding service bulletin is not issued.
- The manufacturer/supplier issues an update, but the airline/MRO mechanic mistakes the unfixed with fixed version and it is not noticed soon enough.

UCA-3.5, 3.6: Manufacturer/Supplier issues service bulletin too late or not at all.

(a) Due to incorrect feedback or incorrect instructions from another controller:

- Not applicable.

(b) Due to process model flaw or algorithm flaw:

- The manufacturer/supplier issues a service bulletin too late or not at all because the internal processes do not ensure that a service bulletin is issued.

(c) Control action is issued but not effective:

- The service bulletin is issued, but it is not received by all airlines because letters are lost in the mail, or an e-mail server did not deliver the e-mail.
- The service bulletin is issued and received by the airline, but the airline does not act on it because the service bulletin does not set a deadline.

4) Loss Scenarios for Airline/MRO Management UCAs

UCA-4.2: Airline/MRO Management does not order mechanics to update software.

(a) Due to incorrect feedback or incorrect instructions from another controller:

- The Airline/MRO management misreads a line in the maintenance log and believes that the update was performed even though it was not.

(b) *Due to process model flaw or algorithm flaw:*

- The Airline/MRO management does not order the mechanics to update the software because only a service bulletin (and not an AD) was received.

(c) *Control action is issued but not effective:*

- The Airline/MRO management orders the airline mechanics to perform the update, but the update is not completed because there are not enough time and personnel available.

5) **Loss Scenarios for Airline/MRO Mechanics UCAs**

UCA-5.1, 5.3: Mechanic does not install or update software per type certificate or SB.

(a) *Due to incorrect feedback or incorrect instructions from another controller:*

- Mechanic installs software onto LRU not as specified by the type certificate because the mechanic was provided the wrong software to install. This may be attackers carried out man-in-the-middle attack and sent modified software.
- Mechanic installs software onto LRU not as specified by the type certificate because the mechanic accidentally installed the wrong software.

(b) *Due to process model flaw or algorithm flaw:*

- Mechanic installs software onto LRU not as specified by the type certificate because the software was modified on the maintenance laptop by attackers.
- Mechanic does not update software according to service bulletin because the mechanic was not following the service bulletin instructions.

(c) *Control action is issued but not effective:*

- Mechanic does not properly carry out the install because he forgets one of multiple steps in the installation process.
- Mechanic installed the wrong software because the loading device contained multiple software versions and the mechanic selected the wrong one.

UCA-5.2: Initial software install not performed when LRU is new.

(a) *Due to incorrect feedback or incorrect instructions from another controller:*

- Mechanic assumed that no install was necessary because he did not get the correct instructions from the airline management.

(b) Due to process model flaw or algorithm flaw:

- Mechanic forgot to install the software because some parts come with software preinstalled while others do not.
- Mechanic did not install the software because he thought another mechanic would do it.

(c) Control action is issued but not effective:

- Mechanic could not carry out the installation because he did not have the right tools for the job.

UCA-5.4: Software update not provided in accordance with SB or AD.

(a) Due to incorrect feedback or incorrect instructions from another controller:

- Mechanic received the service bulletin but did not get software update or proper instructions from airline/MRO management.
- Mechanic did not install the update because he believes that the software was already updated.

(b) Due to process model flaw or algorithm flaw:

- Mechanic decides not to apply the update because he is in a rush and thinks the update is not critical because it's just a SB and not an AD.
- Mechanic intends to do the update but installs the wrong software because it is difficult to tell different versions apart.

(c) Control action is issued but not effective:

- Not applicable.

6) **Loss Scenarios for Pilots UCAs**

UCA-6.1: Pilot does not do pre-flight check when software does not conform with TC.

(a) Due to incorrect feedback or incorrect instructions from another controller:

- Pilot does not perform the pre-flight check because the airline's procedures do not require it.

(b) Due to process model flaw or algorithm flaw:

- ➢ Pilot thinks the pre-flight check was already performed because he was distracted during the process.

(c) *Control action is issued but not effective:*

- ➢ Pilot performs the pre-flight check of LSAP but does not notice that malicious software is installed because the software announces the same version number.

<u>**UCA-6.3, 6.4:** Pilot does not react or reacts too late to alerts.</u>

(a) *Due to incorrect feedback or incorrect instructions from another controller:*

- ➢ Pilot does not react because there is another or conflicting alert issued at the same time.

(b) *Due to process model flaw or algorithm flaw:*

- ➢ Pilot does not react because he/she believes that the alert is just a nuisance alert.

(c) *Control action is issued but not effective:*

- ➢ Pilot reacts to the alert, but it is not effective because another system prevents the control action.
- ➢ Pilot reacts too late to the alert because he/she is distracted when there is imminent unusual and potentially dangerous circumstance.

7) <u>**Loss Scenarios for Pilots UCAs**</u>

<u>**UCA-7.1:** LSAP does not provide alert when dangerous circumstance is imminent.</u>

(a) *Due to incorrect feedback or incorrect instructions from another controller:*

- ➢ LSAP does not issue alert because the aircraft data (i.e., radar altimeter data, airspeed data, etc.) is incorrect.

(b) *Due to process model flaw or algorithm flaw:*

- ➢ LSAP does not issue alert because the database software (i.e., terrain database, navigation database, etc.) is incorrect or outdated.
- ➢ LSAP does not issue alert because an attacker intentionally loaded wrong database.
- ➢ LSAP does not issue alert because an attacker modified the LSAP software to suppress alerts in certain conditions.

(c) Control action is issued but not effective:

- LSAP issues an alert, but the pilot does not see or hear it because he/she is focused on dealing with other alerts/warnings.
- LSAP issues an alert, but the pilot does not see or hear it because there was fault/disconnection on the cables to speakers and warning lights.