System Design Document

For

Aircraft Domain Security Enhancement

Team members: Anthony Johnson, Matthieu Privat, Charles Gilmore, Jacob Stephens, Allen Biagetti, Jorge Santos, Max Gorley, and Max Wilson

Version	**Date**
v1	9/17/21
v1.01	9/24/21
v1.02	9/26/21
v1.03	9/27/21
v1.04	9/28/21
v2	10/25/21
v3	11/30/21
v4	2/3/22
v5	3/6/22
v6	4/7/22

TABLE OF CONTENT

INTRODUCTION **3**

Purpose and Scope 3

Project Executive Summary 3

System Overview 3

Design Constraints 5

Future Contingencies 5

Document Organization 5

Project References 6

Glossary 6

SYSTEM ARCHITECTURE **6**

System Hardware Architecture 7

System Software Architecture 7

Internal Communications Architecture 7

HUMAN-MACHINE INTERFACE **10**

Inputs 11

Outputs 11

DETAILED DESIGN **12**

Hardware Detailed Design 12

Software Detailed Design 12

EXTERNAL INTERFACES **13**

Interface Architecture 14

Interface Detailed Design 14

SYSTEM INTEGRITY CONTROLS **15**

Documentation Security Controls 15

Network Security Controls 15

Penetration Testing Security Controls 16

SYSTEM DESIGN DOCUMENT

Overview
The System Design Document describes the system requirements, operating environment, system and subsystem architecture, files and database design, input formats, output layouts, human-machine interfaces, detailed design, processing logic, and external interfaces.

1 INTRODUCTION

1.1 Purpose and Scope

This document specifies the design for an aircraft domain emulation and testing environment called the Aircraft Domain Security Enhancement version 1.0. This involves the development and configuration of an Aircraft Information Services Domain (AISD) emulation, and a compatible penetration testing kit.

The aircraft domain security enhancement is intended to provide a testing ground for aircraft networks before they are implemented with real networks. There are two stages of development for this project. First, the AISD will be emulated allowing for remote testing and configuration. The emulated network will encompass several medium to low priority components to the network. This will involve several connected software applications specific to the aircraft domain. Second, a penetration testing kit will be developed to test the security of the aircraft's network. The aircraft domain is a complex system containing many communicating critical and non-critical components. Testing these systems becomes increasingly difficult as more components are added which further leads to the importance of a flexible emulated environment. The completion of this project will provide a virtual environment for the security of the aviation domain to be evaluated. This virtual environment will be on a host machine that members of the team can remotely access to run tests on the network.

A greater focus on the system functioning as a CTF has also led to the design of the keyhunts that will be conducted on the system, as well as the design of the interface used to pentest the system with. These two facets will heavily prioritize usability for the pentester. Another focus will be the testing standards used to evaluate security of the network. These will be well defined corresponding to NIST standards to be incorporated into the CTF to accurately evaluate network security. To comply with NIST standards, new expectations of the network include ability to detect and adapt to intrusions.

1.2 Project Executive Summary

1.2.1 System Overview

In the first stage of this project, there are two overarching independent systems. The first system is the AISD network emulation. This emulation is intended for the user to run cybersecurity tests on and reconfigure as needed. The second system is the penetration testing kit that will be implemented specifically for this system. This kit will include all tools necessary to interface with the proprietary systems used in the aircraft domain.

AISD Network Emulation

This test network will consist of four independent nodes that will perform medium to low priority tasks in the AISD. Each machine will be emulated as virtual machines and communicate through a local network. Through running cybersecurity tests, this system will add components to patch security holes so there will be a highly modular approach to implementing this network.

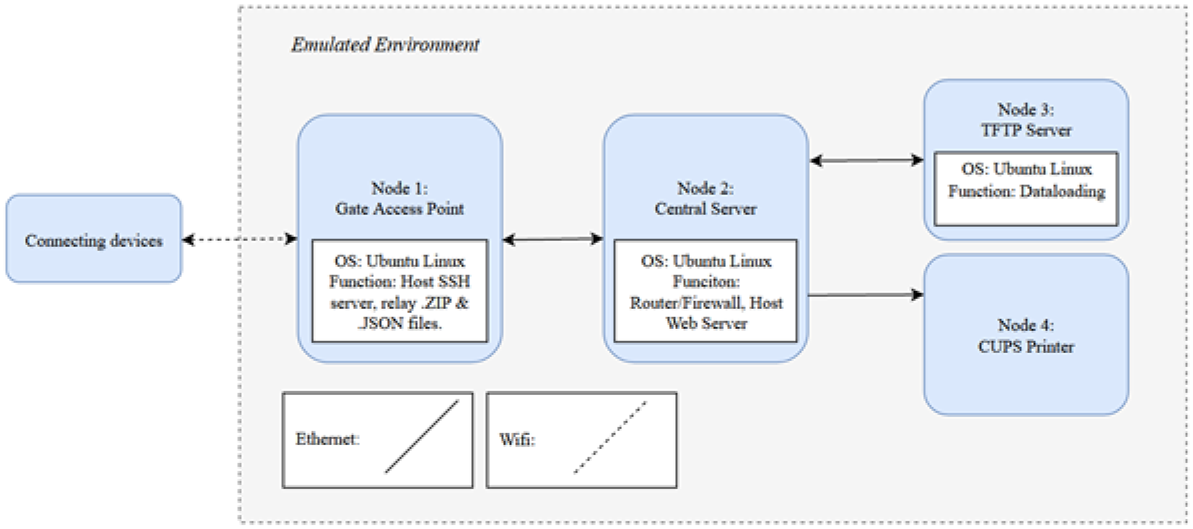


Figure 1: Emulated AISD Figure

Node 1: Gate Access Point

The gate access point is the first point of contact to the network for the users. This machine will run applications that are used to gather needed flight data, update machines, and further interact with the network. It is intended that either a maintenance worker can physically jack into the network to update software or airlines can wirelessly connect and upload files for weather updates and such. With both of these applications, authentication should be required to access communication with this machine and its contents.

Node 2: Central Server

This machine will consist of the AISD network's router, firewall, log server, and web server to access functions of the network. This machine will also include software to implement the 802.1x protocol to further authenticate access in the network. The purpose of this machine is to route and filter network traffic for the AISD network and serve as a bridge to the ACD network.

Node 3: Data Loading Server

This machine will host a TFTP (Trivial File Transfer Protocol) server which communicates with the maintenance application in Node 1 to update machines in the AISD or the ACD networks. Updated packages will be sent through the network to this computer and an application will verify and dispatch updates.

Node 4: Printer

This machine receives instructions from the central server and is used to print files sent to it. In later iterations of this project, the printer might be a physical printer but for now, it will only be emulated.

Penetration Testing Kit

This penetration testing toolkit runs off of a Linux-based security OS that is similar to Kali, which will be loaded with tools needed to conduct a CTF. There are several tools to be built into the system that can be modified to help examine and test the security of the main aircraft network. Several custom scripts were built and added to the toolkit, as well as have been tested on the network. This includes both man-in-the-middle and packet sniffing tools. These tools are used by pentesters to intercept messages sent in the network to uncover keys. The man-in-middle script can be modified for the specific CTF objective the team plans on doing to find the hash or encrypted key.

Another major tool is nmap which can scan the network and see what ports are open during a pentest. The last aspect of the toolkit is to determine how to test the network log and it's vulnerabilities. As of now, research is being conducted on what forms of tools are already out there that could be used or modified based on the needs of the pen tester. During the CTF, there will be a goal that involves getting into the network log and modifying it in some way. These plans are to be flushed out in the next sprint before the final CTF.

1.2.2 Design Constraints

The largest constraint on the system is its hypervisor's capabilities. All of the networking must be managed through the hypervisor which is only a virtual version of routers/switches that would be used to implement this system on an aircraft. Furthermore, the hypervisor may come with some tradeoff in performance. Though hardware acceleration for virtualization can be implemented, there is still some performance tradeoff when emulating the entire network.

1.2.3 Future Contingencies

Due to the uncharted nature of aircraft network penetration testing and multiple obstacles such as funding, hardware issues, and software incompatibility, there are multiple contingencies in place to overcome these problems. The first was switching from a Windows OS to a Linux-based OS which could better accommodate the QEMU virtual machine structure. Requests can be put in at any time to handle funding for purchasing new hardware for the project. If any hardware issues arise, the team will troubleshoot the problem and request assistance from Boeing if it persists over time.

Another major contingency is the security of the network, as well as the user's main home network when using the penetration testing toolkit. The ability of malware to escape the network will be mitigated by several safety features built-in such as a firewall and host-only network connection, however, there is a chance it might still leak onto other systems. If this happens, the network will be scrubbed and rebooted, as well as all affected systems quarantined.

1.3 Document Organization

The software design document is broken down into multiple sections with subsections. The first subject is the project overview, followed by the design of the aircraft network. After contingencies and any issues are discussed, the system architecture is stated, along with external structures and interfaces. Lastly is the system integrity goals. These goals are an important aspect of project integrity and security.

1.4 Project References

Github Repository: GitHub/mwils1426/CS-490
Github Repository for Trojan: GitHub/PushpenderIndia/technowhorse

1.5 Glossary

Term	Definition/Abbreviation
ACD	Aircraft Control Domain
AFDX	Avionics Full-Duplex Switched Ethernet
AISD	Aircraft Information Services Domain
CUPS	Common Unix Printing System
CTF	Capture the Flag
DDOS	Distributed Denial of Service Attack
FMC	Flight Management Computer
HOST-ONLY NETWORK	An isolated network that is cut off from the main internet, but allows access to a set range of IP addresses for a network.
HTTP	Hypertext Transfer Protocol

IPV4	Internet Protocol - Version 4
MITM	Man in the Middle (Attack)
NAT	Network Address Translation
OS	Operating System
QEMU	Virtual OS/network emulator
RAT	Remote Access Trojan
SATCOM	Satellite Communications
SFTP	Secure File Transfer Protocol
SSH	Secure Shell
TFTP	Trivial File Transfer Protocol

2 SYSTEM ARCHITECTURE

The system for this project is a virtual aircraft network emulated using QEMU and Linux-based software. The network consists of multiple "nodes" which are specific parts of the network such as the printer or TFTP server. This system is hosted via a physical laptop/desktop that runs on Linux OS. It acts as the SSH server gateway to access the aircraft network nodes.

2.1 System Hardware Architecture

The current hardware is the host machine which has the emulation software on it, as well as connecting the network through the local router. Future use of airplane components could take place but are not implemented. The use of a cloud network to connect all of the nodes is also under development.

2.2 System Software Architecture

The implementation of newly developed software in the Aircraft Domain Security Enhancement is primarily focused on the penetration testing kit. Our team is using existing software to emulate the network and will focus on developing a kit to interact with said network. Though there are some additional software scripts written to emulate the network, they will not be complex enough to utilize multiple classes. Nonetheless, these scripts will be detailed as subroutines of the system.

2.3 Internal Communications Architecture

The Aircraft Domain Security Enhancement heavily implements networking and uses its network architecture to provide security for both non-critical and critical systems aboard an aircraft. To fully describe the internal communication of the system, this section will cover three levels of communication. The highest level of communication is message passing between applications. This will include files that are passed from machine to machine, or data passed through APIs from program to program. Next, the specific network connections will be described. This will include the emulated hardware components such as routers, firewalls, servers, and etc. Finally, the networking protocols used and their implications on the system and its security will be detailed.

There will be three types of files sent through the network for two applications. The first application is under the airlines' application for the gate access point. Airlines will connect to the network and upload XML or JSON files that include flight and weather data. These files will be stored on the gate access point machine. The next application is maintenance. Maintenance will be able to upload compressed ZIP files that can be used to update parts of the system. In the emulated network, we will focus on these files being sent to the gate access point, then proceed to be sent to the TFTP server. These files will be stored on the TFTP server.

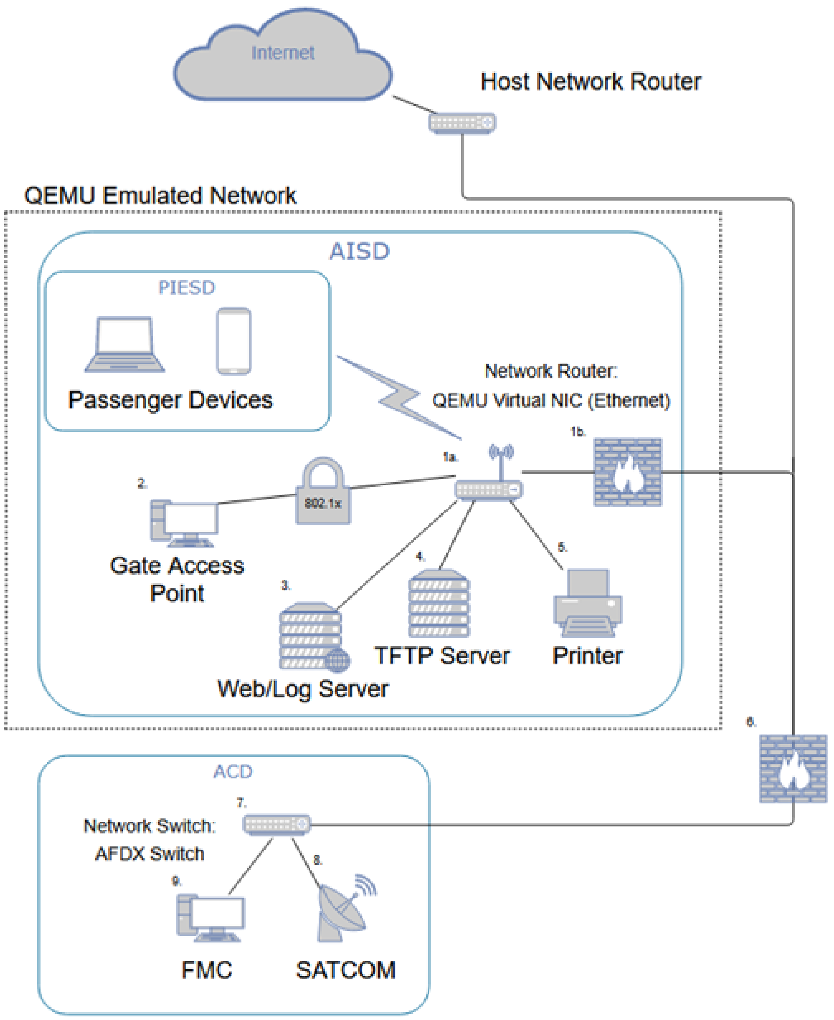


Figure 2: Aircraft Network Security Enhancement Network

Figure 2 Elements Description

2.3.1. AISD Network Router

The network router will be configured through the virt-manager and will allow for the connection of several emulated devices. This network will be the standard ethernet ipv4 network with some forms of encryption to secure transmitted data.

2.3.1.1. AISD Network Firewall

For the network, QEMU has a firewall that will be configured to filter network traffic through the virtual network. This will monitor traffic coming in and out of devices of the network but its scope does not exceed the emulated environment.

2.3.2. Gate Access Point

The gate access point is running Ubuntu and will maintain an authorized connection with the network. This is done through an 802.1x connection and a supplement file that will contain credentials.

2.3.3. Web/Log Server

This server will maintain a network log of the system as well as provide services to devices onboard the aircraft. This will allow passengers to access things such as entertainment and a basic connection to the internet.

2.3.4. TFTP Server

The TFTP data loading server uses functionality provided in the QEMU networking capabilities. Its primary function is to upload files mainly for software updates onto other computers.

2.3.5. Printer

The printer will use the Common UNIX Printing System (CUPS) and will receive messages through Internet Printing Protocol. These messages will be sent through HTTP and can be monitored by the network's firewall.

As previously mentioned, the network contains two main protocols depending on the criticality of the devices connected to that network. The emulated AISD network will be connected through standard ethernet, and the ACD network will be connected through industrial ethernet. The industrial ethernet in this project will be avionics full-duplex switched ethernet which is commonly used in newer aircraft. AFDX switches use this special protocol to provide deterministic timing and redundancy management to provide for a secure network. Because of this, the ACD cannot be currently emulated because this industrial ethernet network is not currently supported under QEMU though this could be a future implementation.

3 HUMAN-MACHINE INTERFACE

The human machine interface can be thought of from two perspectives: one being the red team's pentesting desktop, and the other being the blue team's interface to the log server. For the purpose of this project, the focus on design is with respect to the red team's interface to the system. This is because it is implied that the blue team will act as the system administrators.

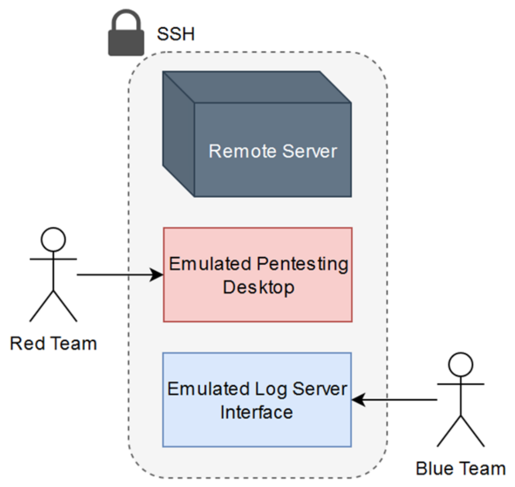


Figure 3: CTF Input Layout

3.1 Inputs

Inputs to this system involve all of the inputs that are possible with connecting to the host machine with an SSH connection. This includes configuring the QEMU network and giving inputs to any of the emulated machines on that network. This interaction is intended to be done through a virtual interface.

Red Team Inputs:
The red team will be using a virtualized desktop and will send inputs via normal keyboard and mouse operations on the desktop. This will include sending keystrokes to execute terminal commands and interact with files on the desktop.

Blue Team Inputs:
The blue team will be primarily controlling the system through the log server which can be operated through a web browser. The blue team will have predefined conditions for network traffic that they can monitor or adjust through the interface.

3.2 Outputs

Because the primary use of this system is for network penetration testing, the outputs will center around logs for devices on the system and the network itself. The log server will continuously log network traffic as well as status updates from machines. These logs are outputs that can be examined after a penetration test. In addition, an output of the display will be streamed to the pentester so it will be easier to interact with the network and the tools available.

4 DETAILED DESIGN

This section contains detailed information regarding the components used for the implementation of the Aircraft Domain Security Enhancement system. Hardware components include all the necessary properties of a system to support the project. Software components include subroutines that must execute throughout the project.

4.1 Hardware Detailed Design

The hardware needed to run the system consists of a Linux-based OS computer and necessary connectivity devices including a router and ethernet cables. The computer needs at least 16 GB of RAM, as well as a 500 GB hard drive.

The choice for hardware has been offloaded to a cloud service as to not have to wait for computer parts and clearance from IT

Our choice in server is a High-Ram server with 1 TB of storage. The high ram is very helpful for managing multiple virtual machines. Documentation surrounding the permissions required from the university for this, as well as exact specifications will be provided in a separate document for the next group if this project is carried on.

4.2 Software Detailed Design

As mentioned before, the software component of the system will be primarily focused on two areas: the penetration testing kit, and the network. Most of the complex software will be written for the penetration testing kit and several scripts and subroutines will be written to automate message passing in the network.

Subroutine 4.2.1: Automated SFTP
This software component is used by the service routines (Maintenance & Airline Services) to perform a Secure File Transfer Protocol or SFTP that allows for the transferring of files onto and off of the network. Upon being run, this component connects to a host's SSH server via IP. It then invokes the SFTP command to be able to securely transfer files.

Subroutine 4.2.2: Packet Sniffer
This software detects traffic on the network between the gate access point and the main system. Once downloaded to the network, it will determine if a packet over the network is TCP, ICMP, HTTP, etc. One can also modify the packet sniffer to enhance it to provide more robust details about the network due to its compatibility with python.

Subroutine 4.2.3: MITM Attack
This software is known as a Man-in-the-Middle script and once on the network acts as a receiver of packets between one system, such as the TFTP server, and another, such as the network log. The script detects packets like the packet sniffer script, but sends all packets meant for the destination IP, to the MITM user instead, before sending it through to its

destination.

Subroutine 4.2.4: Automated TFTP
This software uses Ubuntu as its base operation system with TFTP installed onto it via the command line. The user is able to log onto the Ubuntu VM and send a file via the command line by typing TFTP and inserting the file name, as well as the destination IP address. This will boot a connection to the other system that the user is sending the file to, and the file is transferred to the other system.

Subroutine 4.2.5: DDOS
The DDOS (Distributed Denial of Service attack) is a python file that spawns multiple client threads who each try to take resources from a host server. In this case the DDOS will be configured to disrupt the log server to carry out other parts of the CTF without interruption.

Subroutine 4.2.6: Brute Force
The brute force routine will be used to iterate through captured and intercepted data to uncover keys. This is provided so the pentesting team will not have to connect to the internet to perform any key cracking.

Subroutine 4.2.7: Automated Key verification
This is a script that is important for the completion verification of the CTF. When the penetration testers think they have a key, the way to check will be to input it into this function. This function will store the keys as hashed values so as to not give away the keys in the code. If the key is correct, the function will output “Correct key”. If it is not, the function will output “Incorrect Key”.

Subroutine 4.2.8: Pentesting Kit Setup File
This setup file will be used for possible future iterations of the project to easily setup the pentesting machine. This will be a bash file that downloads all of the necessary packages for the environment to run.

Subroutine 4.2.9: Log Server Setup File
Similar to the Pentesting Kit Setup File, this file will be used to download all necessary packages for the software running on the log server.

5 EXTERNAL INTERFACES

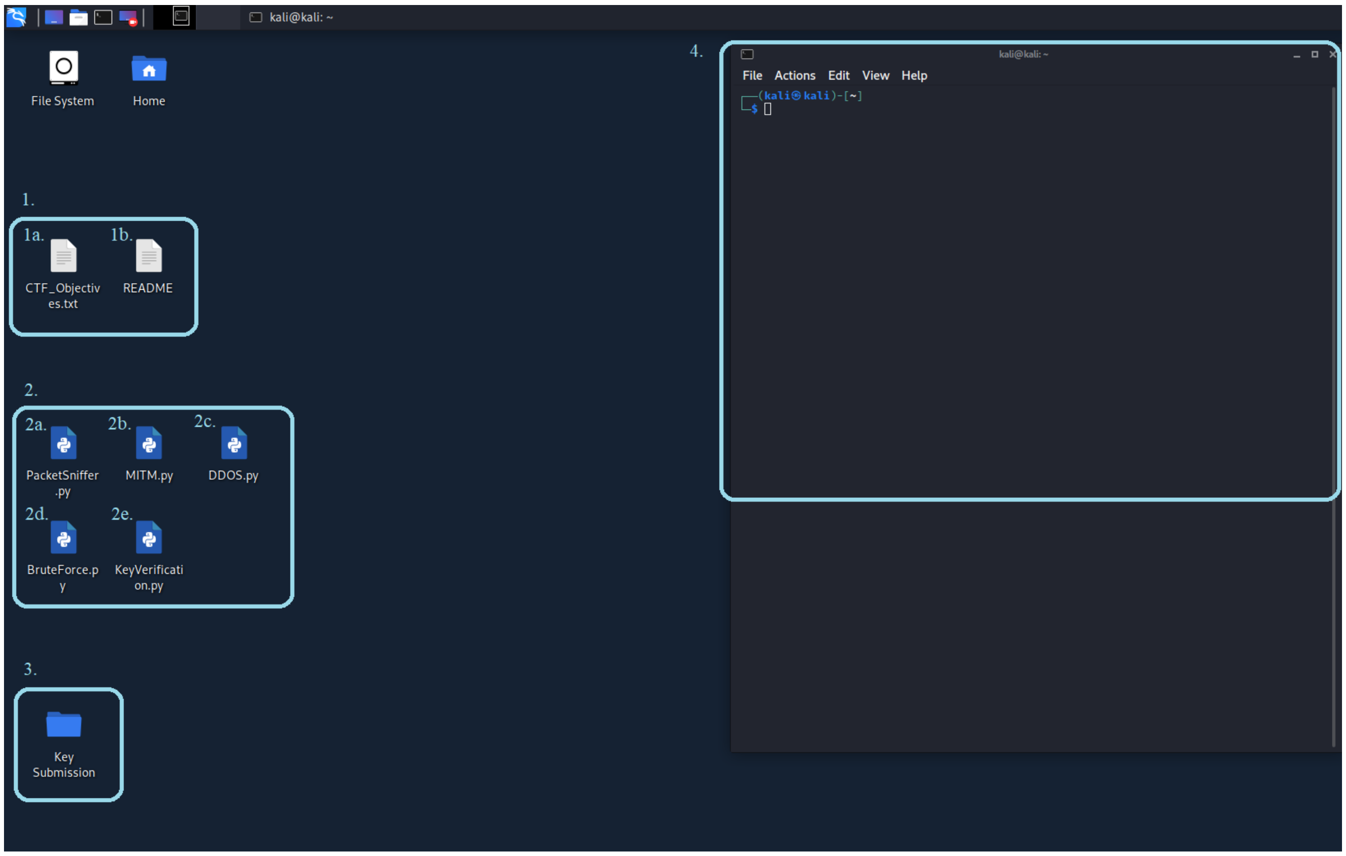
External interfaces include the router of the host computer network, as well as the host computer itself. This host computer is Linux-based and maintains the SSH server and QEMU virtualization of the aircraft network system.

5.1 Interface Architecture

The interface architecture consists of the main host network that is routed through the QEMU virtual machine network so that each part of the aircraft system can interact with one another. Also, the router for the host computer helps maintain the SSH server on the system.

5.2 Interface Detailed Design

As seen in figure 4, the interface will be a virtualized desktop to one computer on the network. For the pentesting team, there will be several files on the desktop that serve as directions, necessary system information, and the conditions for finishing the CTF. The whole CTF will be able to be completed with the files and information provided on the desktop.



```
┌──(kali㉿kali)-[~]
└─$ 
```

Figure 4: Pentesting Interface & Layout

Figure 4 Elements Description

1. CTF Info Files

    This group of files is used to provide the pentesters with the necessary background information to complete the CTF.

    a. CTF_OBJECTIVES.txt

The CTF Objectives file details all of the tasks that must be completed. This includes the keys that need to be captured and the tools that need to be used.

   b. README
      The ReadMe file contains information necessary for operating the environment. This will include all the configurations to run the scripts used in the CTF, as well as necessary network information.

2. Executables
   All executables are files that will be run to perform tasks needed to complete the CTF. These will be missing some small components that the pentesters will need to fill in to target specific nodes or correctly initialize certain tasks. All of the necessary logic will be completed so as to not make CTF overly redundant. All files are described as subroutines in section 4.2
   a. PacketSniffer.py
   b. MITM.py
   c. DDOS.py
   d. BruteForce.py
   e. KeyVerification.py

3. Key Submission Folder
   This is the folder that will be used to store any data from the pentesting team such as captured keys and necessary data for completing the CTF. The contents of this folder can then be inspected and verified.

4. Terminal
   The terminal will be available to the pentesting team to conduct the majority of the networking reconnaissance and execution of subroutines.

6 SYSTEM INTEGRITY CONTROLS

This section contains information on how access to parts of the system will be restricted.

6.1 Documentation Security Controls

The software documentation and information are maintained in an invite-only access drive, with communications securely over a closed-access team server.

6.2 Network Security Controls

The AISD and other nodes of the virtual aircraft network are isolated by a host-only network, as well as have access to a NAT (Network Address Translation). The QEMU system also contains a software-based firewall that is integrated throughout the aircraft network. The host machine for the virtual network also has its own software-based firewall that acts as a buffer between the NAT and host-machine router connection. This allows for more layers of protection over the network, even though each node is connected separately from one another. The TFTP server also has built-in security through the use of only accessing it through SSH.

Only those who are members of the team have access to the network, as well as Boeing as collaborators. The main computer that hosts the SSH server is password locked and requires a team member to connect to it with a password if they would like to access the virtual aircraft network.

6.3 Penetration Testing Security Controls

The penetration testing kit is separated from the rest of the network on a host-only network. This way any malicious activity does not escape from the aircraft virtual network or kit. If any team member uses the penetration testing kit for their own purposes or malicious activity, the team member will be reported to the professor, as well as the dean.