MIL-STD-1553 Intrusion Detection using CUSUM Algorithm

by

Krunal Sachdev

BEng, Gujarat Technological University, 2016

A Report Submitted in Partial Fulfillment

of the Requirements for the Degree of

MASTER OF ENGINEERING

in the Department of Electrical and Computer Engineering

University of Victoria

Supervisory Committee

MIL-STD-1553 Intrusion Detection using CUSUM Algorithm

by

Krunal Sachdev

Bachelor of Engineering, Gujarat Technological University, 2016

Supervisory Committee

Dr. Issa Traore, Department of Electrical and Computer Engineering

Co-Supervisor

Dr. Fayez Gebali, Department of Electrical and Computer Engineering

Co-Supervisor

Abstract

MIL-STD-1553 is a military standard developed by the US department of defense for communication among military avionic platforms (e.g., F-35 and F-16). It has been widely accepted worldwide for more than five decades and is used in many applications other than military avionics. It follows a strict and deterministic procedure for communication among its components. However, research has suggested that it has many vulnerabilities associated with it that can be exploited to carry a range of attacks on it. And since numerous applications make use of this standard, it is crucial to protect MIL-STD-1553 networks.

This project presents an unsupervised anomaly detection scheme using the CUSUM algorithm for the MIL-STD-1553 protocol. A dataset was collected in the ISOT lab by executing six attack vectors on a simulated MIL-STD-1553 network. We leverage the time-based properties of the communication bus to extract a set of relevant features that are fed to the CUSUM algorithm for detection. The experimental evaluation of the proposed detector using the aforementioned dataset yielded promising results, which are very encouraging considering the unsupervised nature of the underlying algorithm.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

- BC: Bus Controller
- BM: Bus Monitor
- CP: Change Point
- CUSUM: Cumulative Sum
- DDoS: Distributed Denial of Service
- DoS: Denial of Service
- DPR: Dual Port RAM
- FCC: Flight Control Computer
- FN: False Negative
- FP: False Positive
- FPR: False Positive Rate
- IDS: Intrusion Detection System
- IRU: Inertial Reference Unit
- ISOT: Information Security and Object Technology
- MAE: Mean Absolute Error
- MCK: Mission Computer Keyboard
- MFD: Multi-function Display Unit
- MIL-STD-1553: Military Standard 1553
- ROC: Receiver Operating Characteristic
- RT: Remote terminal
- TN: True Negative
- TP: True Positive
- TPR: True Positive Rate

Acknowledgments

I want to express my sincere gratitude to my supervisor, Dr. Issa Traore for his continuous support and motivation to pursue my studies and project at the University of Victoria.

I am grateful to Dr. Fayez Gebali for providing the initial motivation towards research and serving on my supervisory committee. In addition, I am greatly indebted to Dr. Issa Traore, who allowed me to join his team as a Research assistant and gave me access to the ISOT laboratory and research facilities. Without his precious guidance, encouragement, and financial support, it would not have been possible to conduct this project.

Chapter 1 : Introduction

1.1 Problem Definition

MIL-STD-1553 is a military standard introduced by the US Department of Defense (DoD) in 1973. It has also been widely used in other branches of armed forces other than military avionics and has served the military for more than 50 years by now. It is based on a master/slave mechanism where the master sends messages in a fixed and predefined time and order. Research conducted by Stan et al. [1] has revealed potential attacks that could compromise the integrity and availability of a MIL-STD- 1553 communication bus. The authors formulated the potential attack vectors and their impacts. Stan et al. [1] also presented an approach for anomaly detection for the MIL-ST-1553 communication bus using the Markov chain model to identify spoofing and DoS attacks on the bus. Spoofing attacks refer to an attack where the attacker gains access to a system by pretending to be someone else to gain confidence in the victim. The DoS refers to a denial-of-service attack where the attacker would send a flood of traffic to make a resource or system unavailable to users. In the DDoS attack, the attacker uses multiple machines to carry out the attack instead of using one machine.

Intrusion detection systems (IDSs) have been around for over four decades, consisting mainly of signature-based and anomaly-based intrusion detection systems. On the one hand signature-based IDSs rely on signatures developed to identify specific attack patterns such as network requests to Command and control servers. As a result, signature-based detection is prone to novel attack methods, such as zero-day attacks. On the other hand, anomaly IDSs rely on normal behavior definitions and observe any activities for deviations from the expected behavior to identify attacks.

Research by Losier [2] shows that the timing properties of the MIL-STD-1553 can be successfully used to detect potential attacks. So, considering the use of MIL-STD-1553 across multiple platforms other than military avionics like commercial platforms, it is imperative to continue research to build an IDS solution for the standard. This project will use the anomaly-based approach to identify attacks.

1.2 Objective and Approach

This report aims to develop an anomaly-based intrusion detection technique based on the CUSUM algorithm to identify attacks on the MIL-STD-1553 communication bus.

At first, we will collect datasets for benign and attack vectors. We will then extract features from the datasets to utilize the known timing properties of the bus, which follow a strict order for communication. After extracting the features, we will use those features as an input to our CUSUM algorithm to identify attacks. This research will be another step in building a complete intrusion detection system for the MIL-STD-1553 standard.

1.3 Report Outline

The remaining chapters in the report are as follows:

Chapter 2 introduces the MIL-STD-1553 standard and its working structure. It also gives a brief overview of the CUSUM algorithm.

Chapter 3 presents the dataset used in the project and the corresponding features extracted from it.

Chapter 4 explains the actual CUSUM algorithm used for anomaly detection and discusses the corresponding experimental results.

Chapter 5 contains the concluding remarks and scope for future work.

Chapter 2 : Background

This chapter gives a brief idea about the architecture of the MIL-STD-1553 standard and gives an overview of CUSUM algorithm for change point detection.

2.1 MIL-STD-1553 Architecture

Figure 2.1 depicts the standard architecture of the MIL-STD-1553 protocol with its primary components: Bus Controller (BC), Remote Terminal (RT) and Bus Monitor (BM).

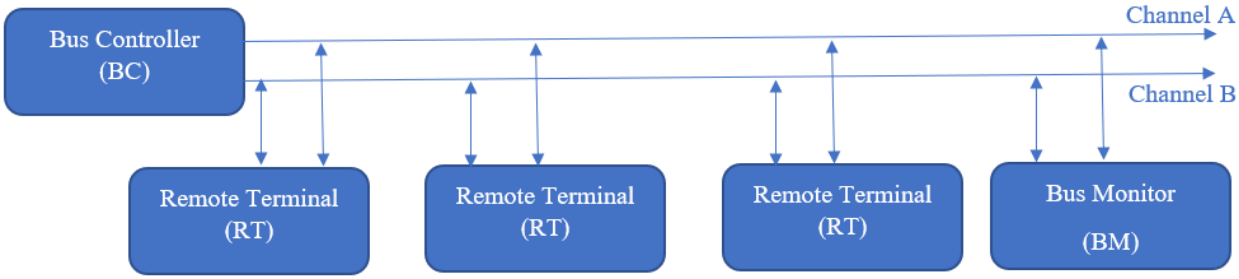

Figure 2.1: MIL-STD-1553 architecture

As mentioned earlier, MIL-STD-1553 uses a master-slave topology to transfer data among its components. Channels A and B are the physical data buses responsible for transferring the data. They both serve the same purpose of being a transmission medium and are dual-redundant. The major components of the architecture are defined below:

- **Bus Controller (BC)**: It is the master in the architecture responsible for initializing the communication between remote terminals (RTs). There might be more than one BC, but only one of them works at a given time. It follows a strict predefined timing and order to send commands to the RT.
- **Remote Terminal (RT)**: It consists of 3 components: subsystem, hardware transceiver and dual port RAM (DPR). The main task of the RT is to transfer data from the subsystem to the bus which BC controls.
- **Bus Monitor (BM)**: It is used for collecting and monitoring data from the bus. It overlooks the operation state and is not responsible for sending any messages.

Words are data structures used for transmitting commands, data, and status over the bus [1]. The communication protocol defines three types of *words*: *command*, *data*, and *status words*. *Command words* are sent by BC to let the RT know if it has to transmit or receive the data. In addition, it contains a few other information like the mode of operation, data word count, mode code and more. The *data word* consists of the actual data transferred between the components. It can follow an arbitrary pattern and is not strictly ordered data. Finally, *status words* are sent by RT to BC to let the master know the current status of an RT. It can also be used to request a service from the BC.

Four major types of communications take place between the components. Figure 2.1 shows the data transfers between the components:

- **BC-RT/RT-BC data transfer:** For BC to RT communication, initially, BC sends a 'receive' command word to RT followed by the actual data words. The RT then acknowledges the data and sends a status word to the BC. On the other hand, in RT to BC communication BC sends a 'transmit' command to RT, which RT acknowledges by sending a status word back. This is followed by the actual data words that RT wants to transmit to BC.
- **RT-RT data transfer:** As shown in figure 2.2, RT1 is the receiving node in this example, and RT2 is the transmitting node. Firstly, BC sends 'receive' and 'transmit' command words to RT1 and RT2, respectively. Then, the 'transmit' command is acknowledged by RT2 by sending a status word that immediately follows the actual data words. In the end, RT1 sends a status word to the BC after the actual data is received.
- **Mode code data transfer:** *Mode codes* are commands sent by BC to RTs to change their mode of operation. The commands can be sent to a specific RT or all the RTs.
- **Broadcast data transfer:** As the name suggests, it is used to send data to all the components in the system. BC does the transmission, and all other components are receivers.

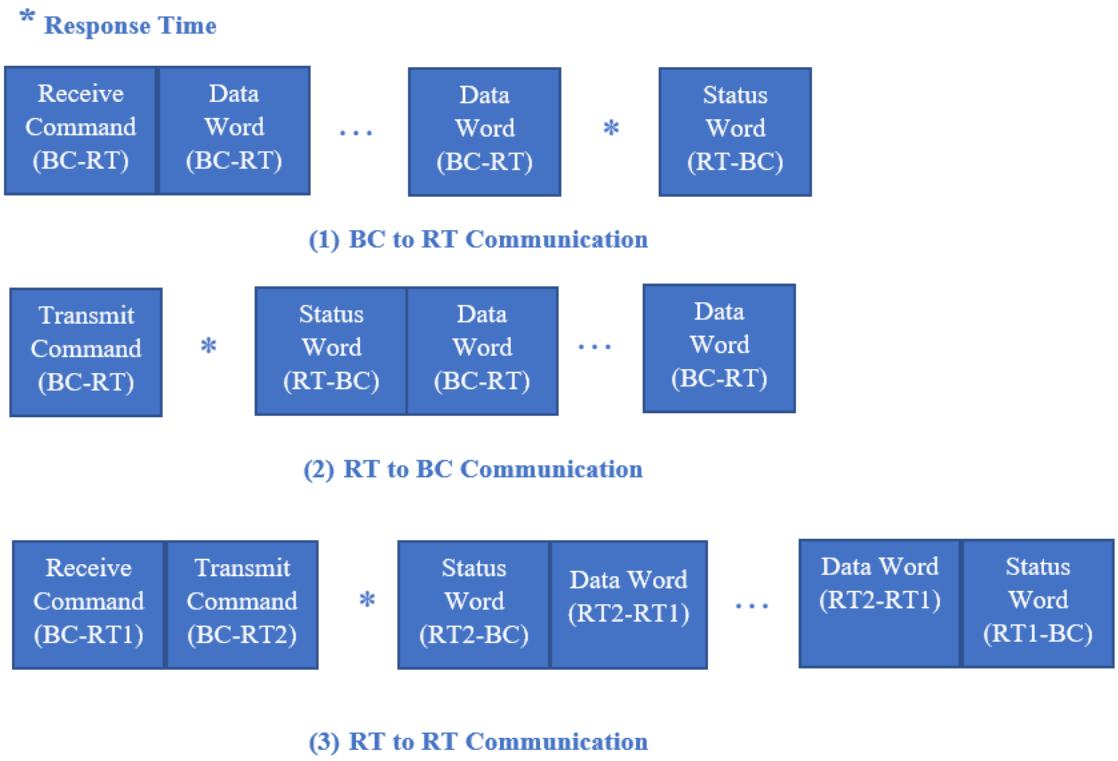

Figure 2.2: Data Transfer formats

2.2 CUSUM Overview

As the name suggests, CUSUM stands for Cumulative Sum. It is statistical sequential change detection algorithm that was discovered by Page at the University of Cambridge in 1954 [3]. It is one of the most well-known algorithms for change detection. Change detection refers to finding changes in sequential data when a property of the time series changes [4]. The first classic CUSUM algorithm was designed for independent and identical distributions.

There are different forms of CUSUM as defined by Page [3]: Direct or recursive forms and one-sided or two-sided forms. The aim of this algorithm, like any other change detection algorithm, is to detect the change when the state of the process changes from normal behavior to an abnormal one at a given time. In other words, a threshold is set for normal behavior, and if that threshold is exceeded, the algorithm sends an alarm at that time.

Chapter 3 : Dataset and Proposed Anomaly Detection Model

This chapter presents the dataset used in our experiments and presents the feature model and detection algorithm involved in the proposed anomaly detection scheme.

3.1 Dataset

The dataset used in this project was collected by the Information Security and Object Technology (ISOT) Lab of the University of Victoria. The architecture used in the data collection is shown in Figure 3.1. The figure shows that the architecture consists of one bus controller and four remote terminals (RT1 – RT4). The bus controller is a mission computer (MC) used to control data transfers between the components. RT1 is an inertial reference unit (IRU) controlled by MC that provides navigation data to the multi-function display and flight control computer, explained below. RT2 is a multi-functional display unit used to display flight data received from other parts of the system such as MC, FCC and IRU. RT3 is a flight control computer (FCC) that generates flight statutes based on navigation and flight path data obtained from the information received by IRU and MC. RT4 is a mission computer keyboard (MCK) used to send data to MC that has been received from the crew. The core buses A and B are used to collect benign data from the normal activities of the bus and malicious data after creating different attack scenarios.

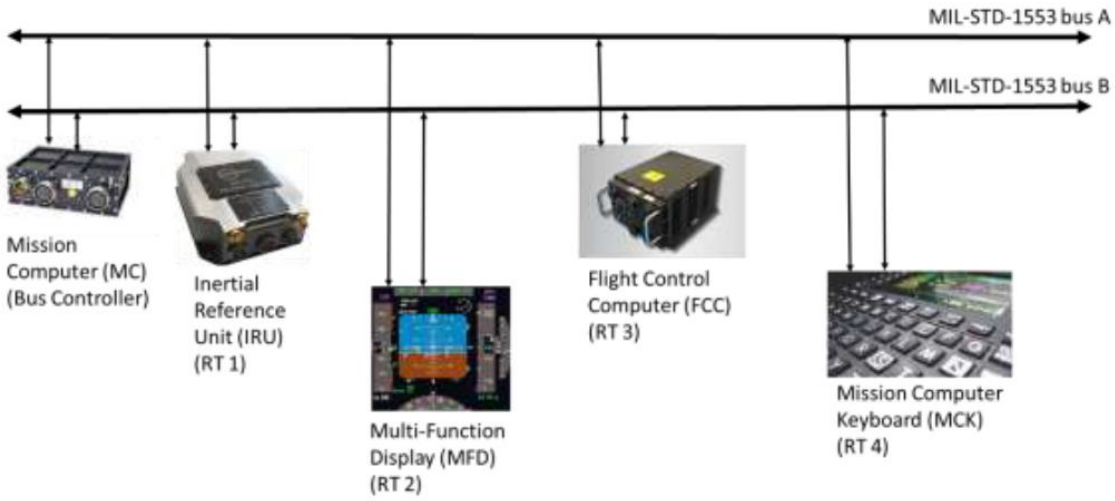

Figure 3.1: Baseline architecture [5]

Twenty-three thousand legitimate messages are generated from the above setup after running it for 10 minutes. The data obtained from the setup was an ASCII dump, so a parser was implemented to convert it to a CSV format for data analysis. Table 3.1 shows the dataset format that contains 55 fields.

Fields	**Description**
msgID	Number associated with the message by the simulator
timestamp	Timestamp of the messages in µs
modeCode	TRUE/FALSE: indicates if the message is a mode command message or not
channel	Channel associated with the message
connType	Type of communication
sa	Address of sending RT
ssa	Sub-address of the sending subsystem from the sending RT
da	Address of receiving RT
dsa	Sub-address of the receiving subsystem from the receiving RT
wc	Word count: total data words in a message
modeCode value	Value when modeCode is set to TRUE
txRsp	Transmit command response time in µs
txSts	Transmit status word
rxRsp	Receive command response time in µs
rxSts	Receive status word
dw0 to dw31	It contains the data words in the message from dw0 to dw32 and will be N/A if there is no data word for a communication
malicious	True/False: Indicator if a particular message is malicious or not.
injected	True/False: Indicator if a particular message is injected or not.
gap	Inter-message gap time in µs
msgTime	Message time in µs

Table 3.1: Dataset Format

3.2 Attack Vectors

In this section, we will talk about the attack vectors that we carried on the simulated MIL-STD-1553 architecture.

We assume that the attacker has gained access to one of the RTs either through manipulating an existing RT or by connecting to it illegitimately. After generating the benign dataset, attack samples were generated by running a total of six 6 different attacks, as seen below:

\

- **Attack 1 (DoS):** This is a DoS attack in which a rogue terminal (RT0) sends random words in a loop to RT3 (MCC). The attack ran for 30 seconds. The data collected contains 148 messages and consists of both benign and malicious data.
- **Attack 2 (Broadcast DoS):** This is also a DoS attack, but, in this case, RT0 sends broadcast messages to all other remote terminals. The attack ran for 30 seconds. The data collected contains 373 messages and consists of both benign and malicious data.
- **Attack 3 (Subtle Injection):** This attack consists of a fake data injection where a rogue terminal (RT0) changes one of the properties of the data (altitude) by replicating one of the messages from a legitimate communication of RT1 (IRU). The attack ran for 30 seconds. The data collected contains 970 messages and consists of both benign and malicious data.
- **Attack 4 (Noisy Injection):** This attack also consists of a fake data injection, but, in this case, RT0 creates fake data. The attack is less noisy and subtle than attack 3, but it can quickly impact the data. The attack ran for 30 seconds. The data collected contains 970 messages and consists of both benign and malicious data.
- **Attack 5 (Logic):** This is a logic attack where a rogue terminal (RT0) sends a broadcast message to shut down the transmitter with a mode code value (0x04). This type of operation is not usual in the middle of communication. The attack ran for 30 seconds. The data collected contains 981 messages and consists of both benign and malicious data.
- **Attack 6:** This is a combination of attacks 4 and 5. The attack ran for 30 seconds. The data collected contains 1004 messages and consists of both benign and malicious data.

3.3 Feature Model

Timing properties are crucial for the operation of the MIL-STD-1533 bus. The communication bus has many timing properties defined in the MIL-STD-1533 standard [6]. Research done by Stan et al. [1] shows that 5 of those properties can be used to detect anomalies on the bus, and it has been confirmed from the research conducted by Losier [2]. The study conducted by Losier [2] used these timings properties and a Histogram approach to detect intrusions on the bus. At least one of the timings properties will detect the intrusion based on its nature. The properties are defined below:

1. **Data Throughput:** The number of data words sent in a given time frame across the communication bus.
2. **Bus Utilization:** The percentage usage of time that the communication bus utilizes in a given time frame.
3. **Periodicity:** The mean time between the BC sending messages to a specific RT for a given time frame.
4. **Inter Message Gap:** The time measured when a BC receives a message's final status and sends the initial command for the next message.
5. **Response Time:** The time required by an RT to respond to a command sent by BC.

All the timings parameters can be extracted for a given time frame and are uniform for all RTs except periodicity. Periodicity cannot be calculated for a given time frame on all traffic. It is used for communication between BC and a specific RT, making the data too complex for our algorithm. Hence, periodicity has not been taken into consideration here. Also, the property response time has not been extracted in this work, but a new helpful property based on the modecode value of the dataset has been used. The time frame selected in this work is 200 ms, proving to give the best results compared to the 50 ms used in the previous work [7]. The functions used to extract the properties are explained below:

1. **Data Throughput:** The feature is extracted by summing the word count column for a span of 200 ms.

2. **Bus Utilization:** After getting an individual 200 ms frame, the difference between timestamps is calculated and added, which gives the total time when the bus is not utilized. It is then subtracted from the actual time frame, that is, 200 ms, to find the utilization of the bus in percentage.
3. **Inter Message Gap:** This property was extracted from the dataset itself, but it has been used with a minor tweak in the data for use in the algorithm.
4. **ModeCode:** This feature is extracted for a specific logic attack in which a rogue RT terminal sends unwanted signals in the middle of the operation. The mode code commands sent by RT to BC are extracted and compared with the normal operation.

All the features are normalized by scaling them to the range [0,1]. These properties have been used in the CUSUM algorithm to identify the attacks.

3.4 Intrusion Detection Approach

The algorithm to identify attacks based on CUSUM is written in Python programming language and uses one of the known Python modules for change detection in it.

$$\begin{cases} s[t] = x[t] - x[t-1] \\ g^{+}[t] = \max\left(g^{+}[t-1] + s[t] - drift, 0\right) \\ g^{-}[t] = \max\left(g^{-}[t-1] + s[t] - drift, 0\right) \end{cases}$$

$$if\ g^{+}[t] > threshold\ OR\ g^{-}[t] > threshold$$

$$\begin{cases} t_{alarm} = t \\ g^{+}[t] = 0 \\ g^{-}[t] = 0 \end{cases}$$

Equation 3.1 CUSUM algorithm [8]

The algorithm takes three inputs: the first is the data *x*, the other two are *threshold* and *drift* values. There are many ways to implement the CUSUM algorithm. One of the ways is to find out positive ($g^{+}[t]$) and negative ($g^{-}[t]$) changes in the values of data (x) and then calculate its cumulative sum. The cumulative sum is then compared to the threshold value. If the cumulative sum at a given

time *t* exceeds the threshold, it is made to start from zero, and an alarm(*t*) is generated to detect the change. It can be seen in the above Equation 3.1. The drift, which is another essential parameter, is used here to reduce the false positives. Therefore, the CUSUM algorithm mainly relies on tuning threshold and drift parameters.

According to Gustafsson (2000) [9], the following steps can be used for tuning purposes:

- Choose to begin with a high *threshold.*
- Pick a *drift* value that is half of the expected change or such that $g = 0$ more than half of the time.
- The next step would be to set a *threshold* to obtain detections.
- Decrease the *drift* to achieve faster detection
- Increase the *drift* to reduce false positives.
- The *drift* can also be increased in cases where the changes do not make sense.

The output of the algorithm gives an array of indexes where the changes are detected and their amplitude. This means it provides the location of the data frames in the form of an array to identify the attacks. The output also plots a graph of the detected changes and the data frames.

Chapter 4: Experimental Evaluation

This chapter outlines the evaluation procedure and metrics and present the experiment results.

4.1 Procedure and Metrics

Since the dataset is a collection of datasets consisting of specific attack, we present the experimental results of the algorithm for individual attacks. By comparing the output against the available dataset, we identify the false positives and false negatives. We also plot a detection graph depicting the changes detected for the timeline with its amplitude for the output. Finally, we compute the performance based on the below parameters and plot receiver operating characteristic (ROC) curves for the values:

Accuracy: It is the ratio of the predicted change points to the total data points.

$$Accuracy = \frac{True\ Postive + True\ Negative}{Total\ number\ of\ data\ points}$$

False Positive Rate (FPR): It is defined as negative accuracy. It is also known as specificity.

$$FPR = \frac{False\ Positives\ (FP)}{Number\ of\ Negatives\ (FP + TN)}$$

True Positive Rate (TPR): It is defined as the ratio of true positive change points to total classified change points. It is also known as precision.

$$TPR = \frac{True\ Positives\ (TP)}{Number\ of\ Positives\ (TP + FN)}$$

Mean Absolute Error (MAE): It is defined as the ratio of the absolute value of the difference between the predicted and actual change points to total change points.

$$MAE = \frac{\sum_{i=1}^{\#CP} |Predicted\ CP - Actual\ CP|}{\#\ CP}$$

4.2 Experiment Results

Figures 4.1 and 4.2 show the detection graph and ROC curve for the first Basic DoS attack. As seen in Table 4.1, the data throughput feature performs better than other features. The threshold and drift selected for attack are 0.82 and 0.1, respectively.

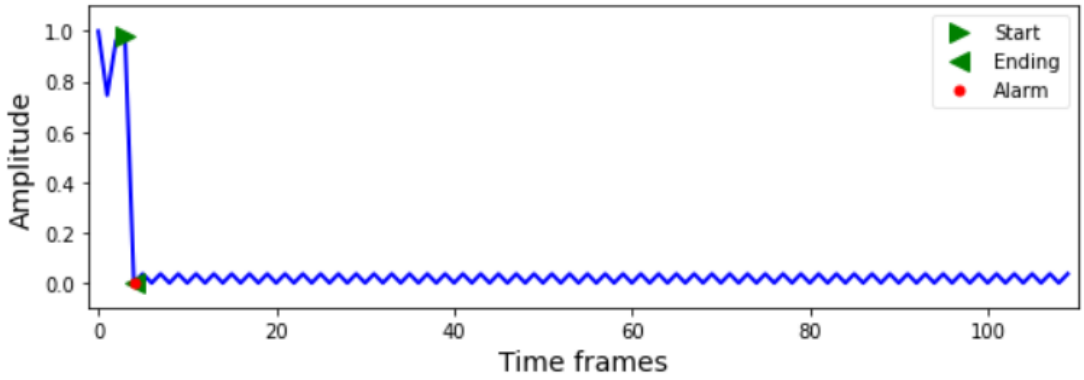

Figure 0.1: CUSUM detection graph: Attack 1(Basic DoS) with Data Throughput feature

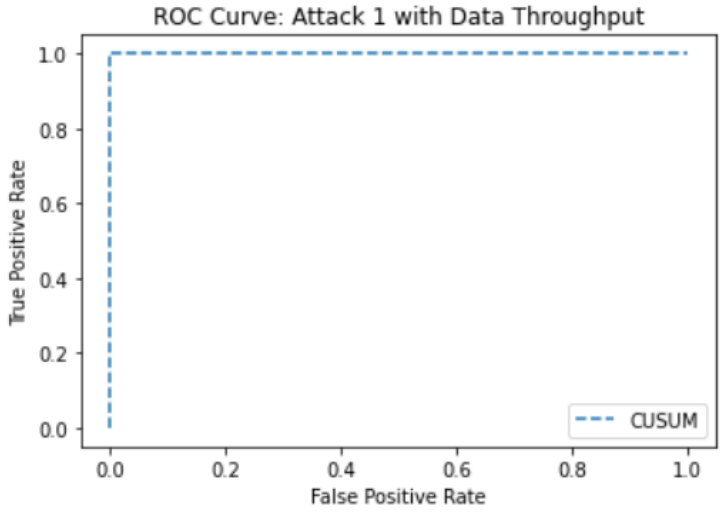

Figure 0.2: ROC Curve: Attack 1 with Data Throughput

Attack 1: DoS Attack (Threshold: 0.82 and Drift: 0.1)				
	Accuracy (%)	**FPR (%)**	**TPR (%)**	**MAE (%)**
Data Throughput	100.0	0.0	100.0	0.0
Inter Message Gap	83.33	100.0	100.0	16.67
Bus Utilization	85.71	0.0	50.0	14.29
Mode Code	83.33	100.0	100.0	16.67

Table 0.1: Performance Evaluation: Attack 1

Figures 4.3 and 4.4 show the detection graph and ROC curve for Attack 2 (Broadcast DoS). Table 4.2 shows that the data throughput feature performs better than other features in this attack scenario. The threshold and drift selected for attack are 0.81 and 0.1, respectively.

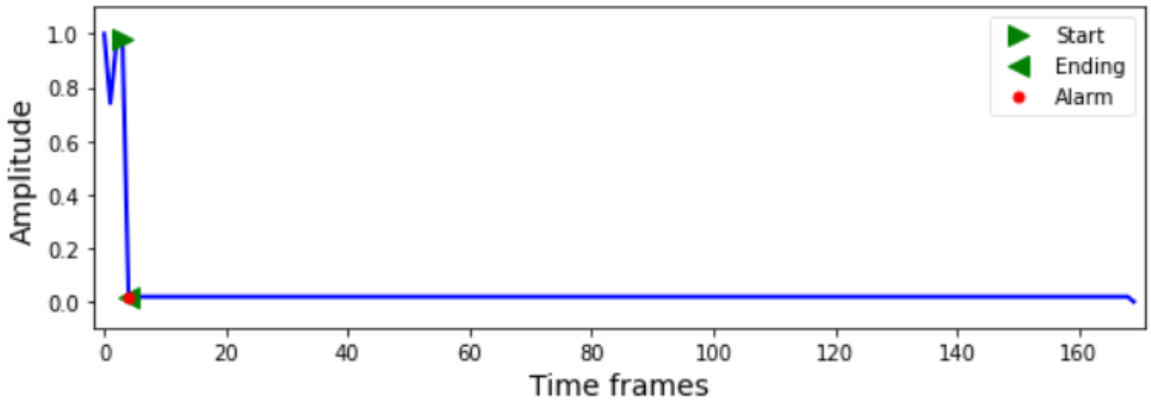

Figure 0.3: CUSUM detection graph: Attack 2(Broadcast DoS) with Data Throughput feature

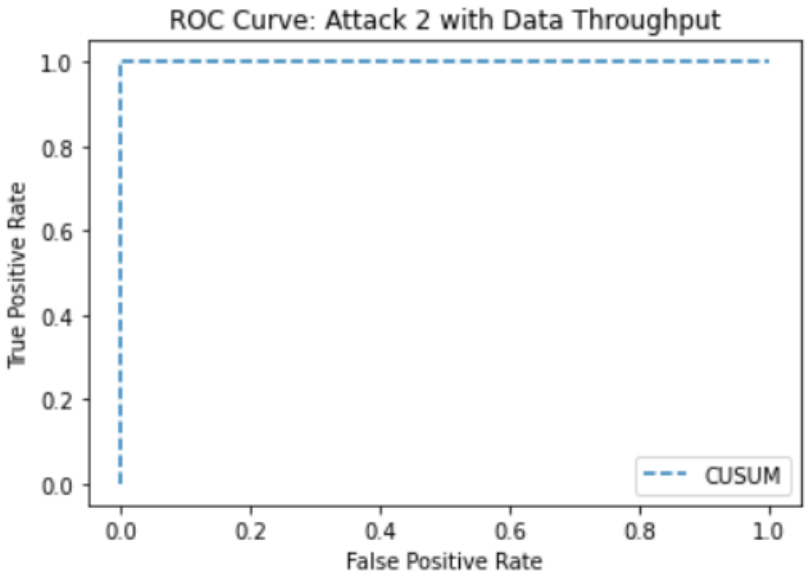

Figure 0.4: ROC Curve: Attack 2 with Data Throughput feature

Attack 2: Broadcast DoS Attack (Threshold: 0.81 and Drift: 0.1)				
	Accuracy (%)	FPR (%)	TPR (%)	MAE (%)
Data Throughput	100.0	0.0	100.0	0.0
Inter Message Gap	66.67	100.0	100.0	33.33
Bus Utilization	80.0	10.0	100.0	20.0
Mode Code	66.67	100.0	100.0	33.33

Table 0.2: Performance evaluation: Attack 2

Inter message gap feature outperforms all other features in Attack 3 (Subtle injection), as seen in Table 4.3. The threshold and drift selected for attack are 0.9 and 0.1, respectively. Figures 4.5 and 4.6 show the detection graph and ROC curve, respectively.

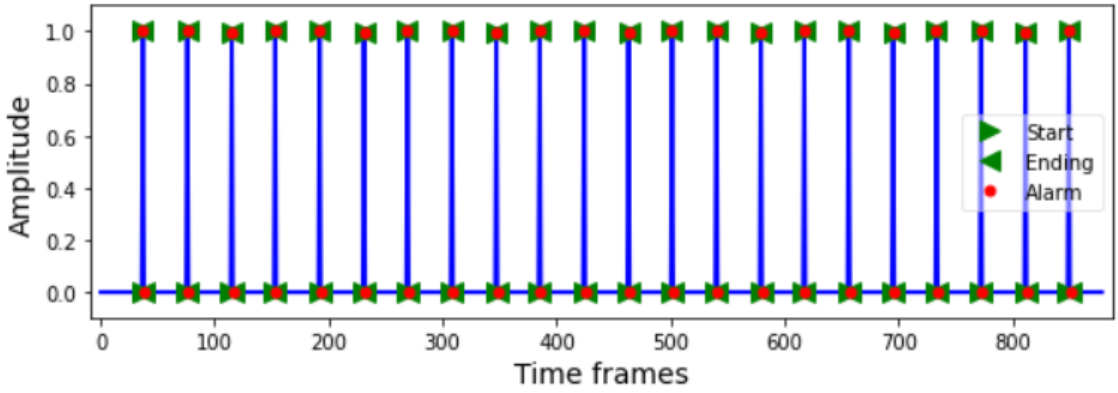

Figure 0.5: CUSUM detection graph: Attack 3(Subtle Injection) with Inter Message Gap feature

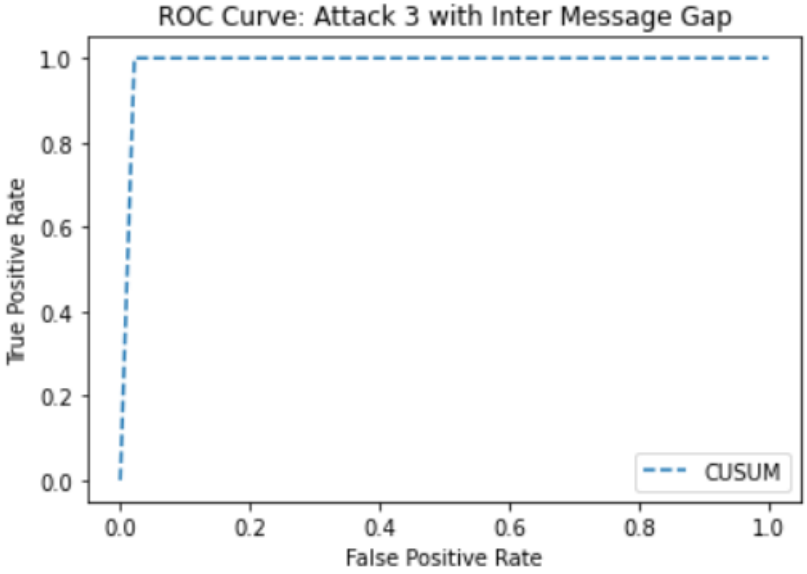

Figure 0.6: ROC Curve: Attack 3 with Inter Message Gap feature

Attack 3: Subtle Injection Attack (Threshold: 0.9 and Drift: 0.01)				
	Accuracy (%)	FPR (%)	TPR (%)	MAE (%)
Data Throughput	59.29	39.639	0.0	40.71
Inter Message Gap	97.87	2.17	100.0	2.13
Bus Utilization	45.39	55.07	66.67	54.61
Mode Code	97.85	100.0	100.0	2.15

Table 0.3: Performance evaluation: Attack 3

Figures 4.7 and 4.8 show the detection graph and ROC curve for Attack 4 (Noisy injection). The threshold and drift selected for attack are 0.9 and 0.01, respectively. Table 4.4 shows that the inter message gap feature performs better than other features in this scenario.

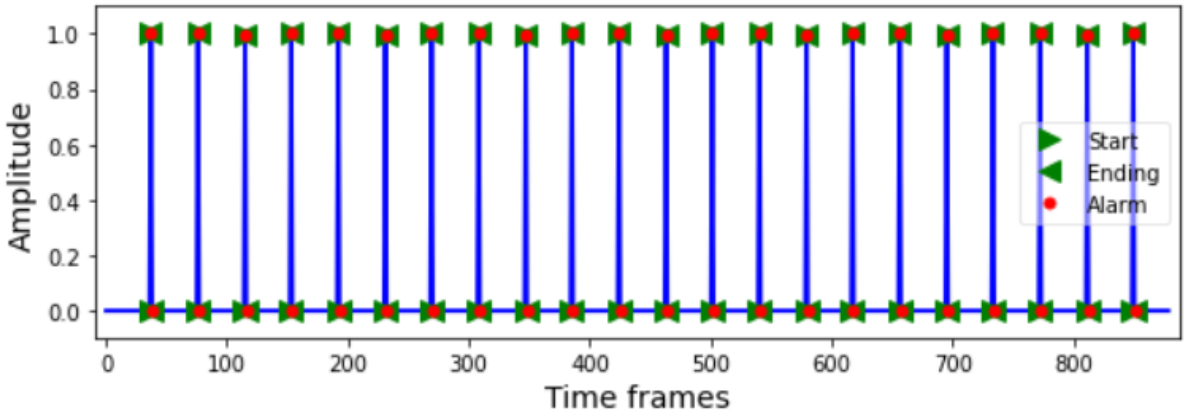

Figure 0.7: CUSUM detection graph: Attack 4(Noisy Injection) with Inter Message Gap feature

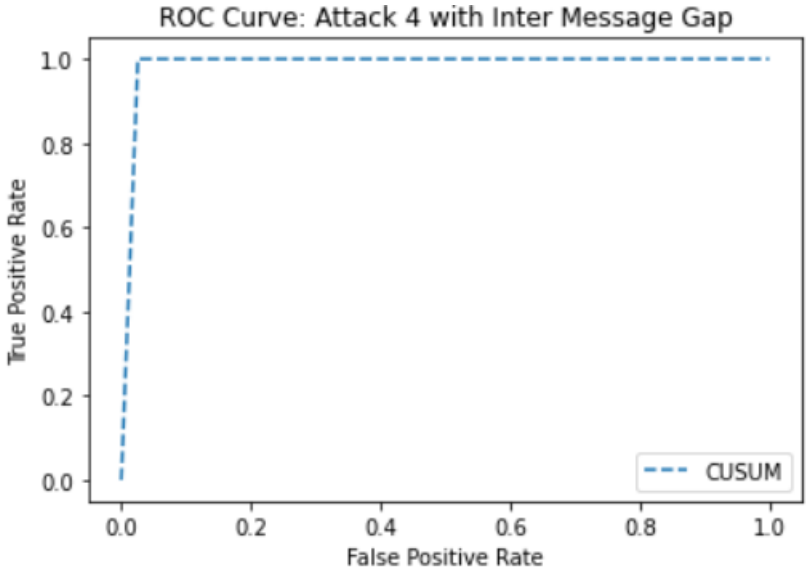

Figure 0.8: ROC Curve: Attack 4 with Inter Message Gap feature

Attack 4: Noisy Injection Attack (Threshold: 0.9 and Drift: 0.01)				
	Accuracy (%)	FPR (%)	TPR (%)	MAE (%)
Data Throughput	60.17	38.73	0	39.82
Inter Message Gap	97.50	2.56	100	2.50
Bus Utilization	45.39	55.07	66.67	54.61
Mode Code	97.87	100	100	2.13

Table 0.4: Performance evaluation: Attack 4

As seen in Table 4.5, the ModeCode feature outperforms all other features for Attack 5 (Logic attack). The threshold and drift selected for attack are 0.9 and 0.01, respectively. Figures 4.9 and 4.10 show the attack's detection graph and ROC curve.

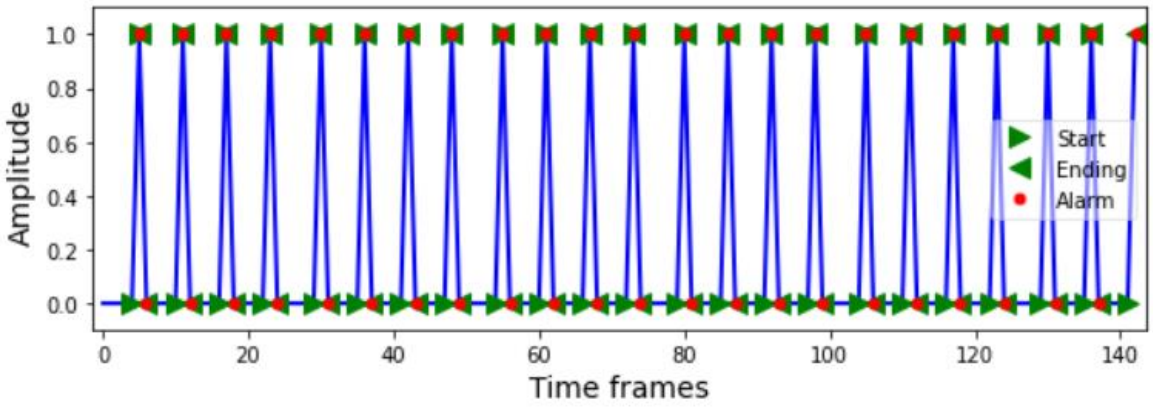

Figure 0.9: CUSUM detection graph: Attack 5(Logic) with ModeCode feature

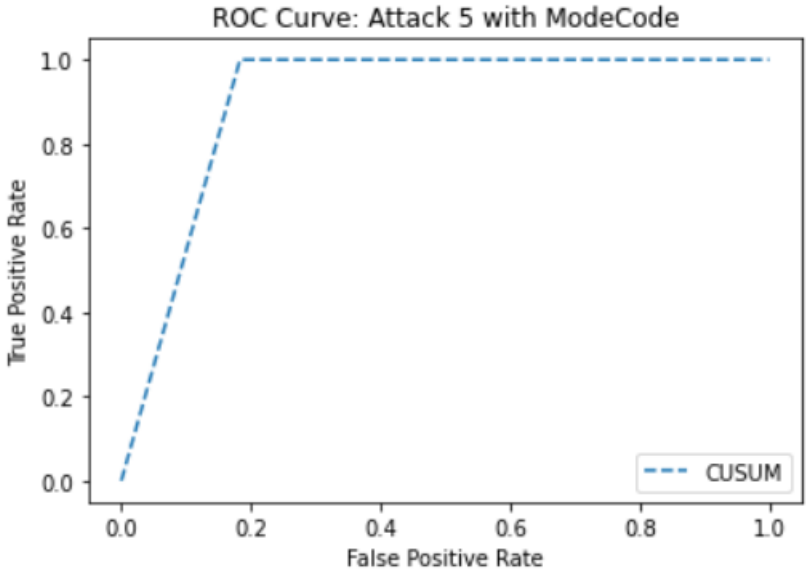

Figure 0.10: ROC Curve: Attack 5 with ModeCode feature

Attack 5: Logic Attack (Threshold: 0.9 and Drift: 0.01)				
	Accuracy (%)	FPR (%)	TPR (%)	MAE (%)
Data Throughput	53.91	41.23	27.78	46.09
Inter Message Gap	92.35	5.20	0.0	7.65
Bus Utilization	52.45	51.67	73.91	47.55
Mode Code	84.62	18.33	100.0	15.38

Table 0.5: Performance evaluation: Attack 5

Figures 4.11 and 4.12 show the detection graph and ROC curve for Attack 6 (Combination attack). Table 4.6 shows that the inter message gap feature performs better than other features in this attack scenario. The threshold and drift selected for attack are 0.9 and 0.01, respectively.

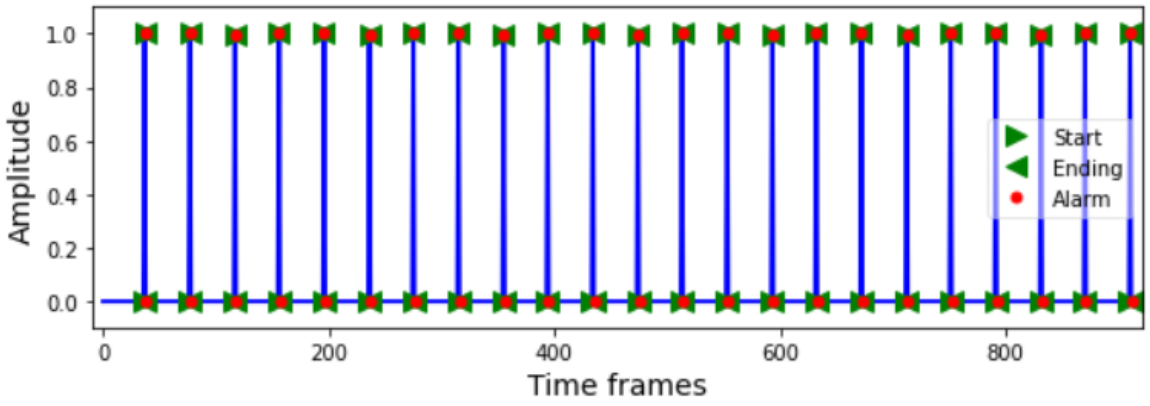

Figure 0.11: CUSUM detection graph: Attack 6(Combination) with Inter Message Gap feature

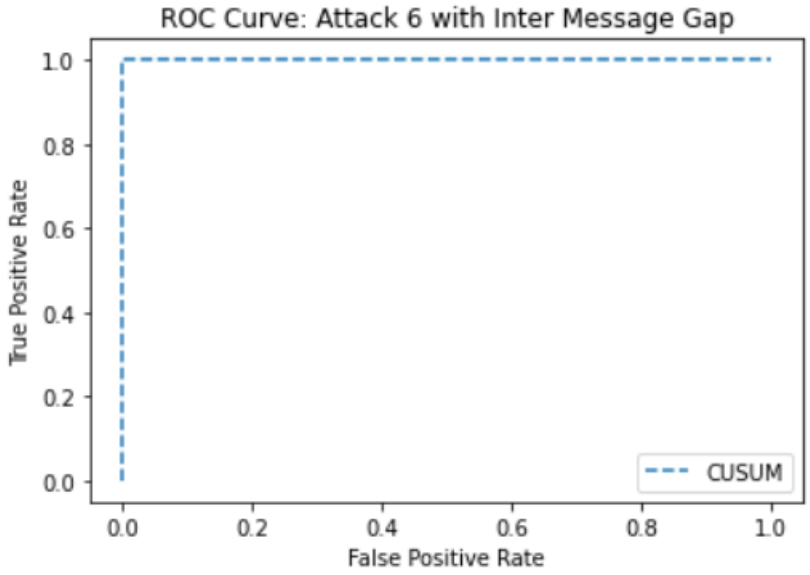

Figure 0.12: ROC Curve: Attack 6 with Inter Message Gap feature

Attack 6: Combination Attack (Threshold: 0.9 and Drift: 0.01)				
	Accuracy (%)	FPR (%)	TPR (%)	MAE (%)
Data Throughput	93.04	3.6	0.0	6.96
Inter Message Gap	100.0	0.0	100.0	0.0
Bus Utilization	56.64	40.87	0.0	43.36
Mode Code	68.53	30.65	50.0	31.47

Table 0.6: Performance evaluation: Attack 6

4.3 Results Discussion

The above results show that the data throughput will help detect attacks similar to Denial-of-Service (DoS) attacks. The basic DoS is carried by a rogue terminal sending random words to a specific RT. While, in the broadcast DoS attack, the rogue terminal sends broadcast messages to all other remote terminals. The inter message gap feature is better suited in attacks where fake data injection occurs. The rogue terminal manipulates either a property value or the entire message while sending the data. It is evident from the above findings that the subtle injection attack where rogue terminal changes one of the message properties and the noisy injection attack where the rogue terminal changes entire data are detected with high accuracy and low FPR rate. Logic attacks are detected efficiently by the mode code feature. These attacks are carried out by sending random commands like shutdown in the middle of regular communication. The last attack, a combination of logic and fake data injection, was detected accurately with the Inter message gap property.

These findings show that the CUSUM algorithm's primary objective is to detect the attacks as efficiently as possible. However, to build a complete IDS, the above results should use a certain combination of features to detect attacks. This can be done by combining more than one feature to detect the anomaly using logical operators. The performance properties like accuracy, FPR, TPR from the output from the algorithm can be used to implement a logic further and make the technique more robust. So, this work brings us a step closer to building a complete IDS for the MIL-STD-1553 standard.

Chapter 5: Conclusion and Future work

MIL-STD-1553 is a widely accepted military standard across various platforms worldwide. However, it has many attack vectors associated with it and could have devastating results if exploited. This project aimed to develop an intrusion detection technique for the MIL-STD-1553 standard using the CUSUM algorithm for change detection.

This aim was achieved by using the timing properties of the protocol and extracting four features based on it. Then, a CUSUM algorithm was implemented to use the extracted features and detect six attacks on the MIL-STD-1553 communication bus. Finally, performances and tests were carried out to validate the algorithm, and the results show that individual features gave very positive results to detect specific attacks. This proves that the standard's time-based properties can effectively detect such intrusions.

Future work on this could include using additional properties like periodicity and response time, which have proven promising in the past [2]. In addition, as discussed in the results section of the report, future work would include implementing some logic based on the output of the features to identify attacks and make the technique more robust. This would include working on the performance parameters, applying a combination logic using the logic operators (e.g. AND, OR, etc.), and using the additional properties discussed earlier. Though the technique presented in this project successfully detects six attacks, it does not cover all attacks, and more work can be done to include newer attack vectors for the bus. Also, the CUSUM technique used in this project brings us a step closer to building a complete IDS for the MIL-STD-1553 standard.

References

[1] O. Stan, Y. Elovici, A. Shabtai, G. Shugol, R. Tikochinski and S. Kur, "Protecting Military Avionics Platforms from Attacks on MIL-STD-1553 Communication Bus," 2017.

[2] R. S. a. V. R. B. Losier, "Design of a time-based intrusion detection algorithm for the MIL-STD-1553," Kingston, ON, Canada, 2019.

[3] E. S. Page, "Continuous Inspection Schemes," *Biometrika,* vol. 41, p. 100, 1954.

[4] Y. Kawahara and M. Sugiyama, "Sequential change-point detection based on direct density-ratio estimation," *Statistical analysis and data mining,* vol. 5, pp. 114-127, 2012.

[5] P. Q. H. S. K. S. Issa Traore, "Early Cyber Warning and Response System for MIL-STD-1153 Platforms using Unsupervised Anomaly Detection," Information Security and Object Technology (ISOT) Lab, Victoria, January 2021.

[6] C. deLong, "MIL-STD-1553B Digital Time Division Command/Response Multiplex Data Bus," 2 ed., R. Zurawski, Ed., CRC Press, 2015, pp. 43-1-43-25.

[7] M. Thibault, C19C B1 mission computer (MC) timing and load analysis, CAE Military Support Programs, May 2008.

[8] M. Duarte, "CUSUM Detection," 03 April 2021. [Online]. Available: https://nbviewer.org/github/demotu/detecta/blob/master/docs/detect_cusum.ipynb. [Accessed 13 February 2022].

[9] F. Gustafsson, Adaptive filtering and change detection, Chichester, England;New York;: Wiley, 2000.

[10] M. Basseville and I. V. Nikiforov, Detection of Abrupt Changes: Theory and Application, Prentice Hall, 1993.

[11] S. Aminikhanghahi and D. J. Cook, "A survey of methods for time series change point detection," *Knowledge and information systems,* vol. 51, pp. 339-367, 2016;2017;.

[12] S. J. J. Genereux, A. K. H. Lai, C. O. Fowles, V. R. Roberge, G. P. M. Vigeant and J. R. Paquet, "MAIDENS: MIL-STD-1553 Anomaly-Based Intrusion Detection System Using Time-Based Histogram Comparison," *IEEE transactions on aerospace and electronic systems,* vol. 56, pp. 276-284, 2020.

[13] F. Onodueze and D. Josyula, "Anomaly Detection on MIL-STD-1553 Dataset using Machine Learning Algorithms," in *2020 IEEE 19th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*, 2020.