MIL-STD-1553 Intrusion Detection using ECP e-divisive Algorithm

by

Dhruvil 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 ECP e-divisive Algorithm

by

Dhruvil Sachdev

Bachelor of Engineering, Gujarat Technological University, 2016

Supervisory Committee

Dr. Issa Traore, Department of Electrical and Computer Engineering
Supervisor

Dr. Fayez Gebali, Department of Electrical and Computer Engineering
Supervisor

Abstract

MIL-STD-1553 is a military standard that defines the interface and protocol for data buses used in military aircraft and other aerospace applications. It is a reliable and robust communication protocol that ensures the efficient transfer of data on the bus. However, the increasing use of MIL-STD-1553 in critical systems has led to a growing concern about the security of these systems. Intrusion detection is an important aspect of securing these systems and preventing unauthorized access.

This report presents an approach to detect intrusions on MIL-STD-1553 bus systems by tracking changes with respect to known timing properties of the bus using the ECP e-divisive algorithm. The ECP e-divisive algorithm is a statistical method that is based on the concept of change point detection.

The proposed detection model was evaluated using a dataset collected through a MIL-STD-1553 simulator. The outcome of the evaluation showed that the ECP e-divisive algorithm was effective in detecting intrusions on MIL-STD-1553 bus systems. It can detect a wide range of attack vectors and is also able to adapt to changing conditions on the bus, such as changes in the number of Remote Terminals or the amount of data being transferred.

Table of Contents

List of Tables

List of Figures

Glossary

BC: Bus Controller

BM: Bus Monitor

CP: Change Point

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 am deeply appreciative of the support and guidance provided by Dr. Issa Traore, throughout my studies and research project at the University of Victoria.

I would also like to extend my gratitude to Dr. Fayez Gebali for his role in inspiring me to pursue research and for serving on my supervisory committee.

Dedication

Dedicated to my parents, brother, and friends for their support, motivation and guidance.

Chapter 1 : Introduction

1.1 Context

An Intrusion Detection System (IDS) is a system that is employed to detect and monitor malicious or suspicious activities on a network or on the hosts deployed on the network. IDSs have two primary detection methods: signature-based and anomaly-based. Signature-based IDS detects suspicious activities by recognizing known malicious patterns, also known as attack signatures. On the other hand, anomaly-based IDS builds a model of normal behavior of the network and its entities, and flag deviations or anomalies are possible intrusions. Unlike signature-based detection models, anomaly detection models have greater potential to detect unknown suspicious activities, however, they are subject to a much greater number of false positives.

MIL-STD-1553 is a military standard created by the US Department of Defence (DoD) for military avionics integration in mission-critical systems. It has been in use by various branches of the armed forces since 1973. This standard operates on a master/slave methodology, where the master sends data and messages in a fixed, predefined time and order. Studies have shown the impact of successful cyber-attacks on aerospace vehicles that use the MIL-STD-1553 communication bus [1]. Additionally, [2] presented a security analysis of a supervised, sequence-based anomaly detection method for identifying cyber-attacks such as denial of service (DOS) and spoofing attacks on MIL-STD-1553 systems.

1.2 Objective and Approach

This report endeavors to create an intrusion detection method utilizing the Early Change Point (ECP) e-divisive algorithm to detect attacks on the MIL-STD-1553 communication bus.

An existing dataset that was previously collected in a testbed at the Information Security and

Object Technology (ISOT) Lab was used. The dataset contains both benign samples and attack samples based on different attack vectors. The process includes extracting relevant features from the datasets based on the known timing properties of the bus and inputting these features into an algorithm to identify Denial of Service (DOS), Broadcast DOS, subtle injection, noisy injection, and logic attacks. The goal of this research is to contribute to the development of a comprehensive intrusion detection system for the MIL-STD-1553 standard.

1.3 Report Outline

The report is structured as follows:

Chapter 2 introduces the MIL-STD-1553 standard and its communication structure. It also includes a brief explanation of the ECP package.

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

Chapter 4 delves into the implementation of the ECP e-divisive algorithm for intrusion detection and the analysis of the experimental results.

Finally, Chapter 5 concludes the report with a summary of the findings and suggestions for future work.

Chapter 2 : Background

This section provides a general understanding of the MIL-STD-1553 architecture and an overview of the ECP method for identifying change points.

2.1 MIL-STD-1553 Architecture

MIL-STD-1553 defines a dual redundant serial communication bus for transmitting data between a bus controller and remote terminals following master/slave methodology [2]. As shown in Figure 2.1, it consists of a primary bus (Channel A), with the secondary bus (Channel B) as a fallback, and components: Bus Controller (BC), Remote Terminal (RT), and Bus Monitor (BM).

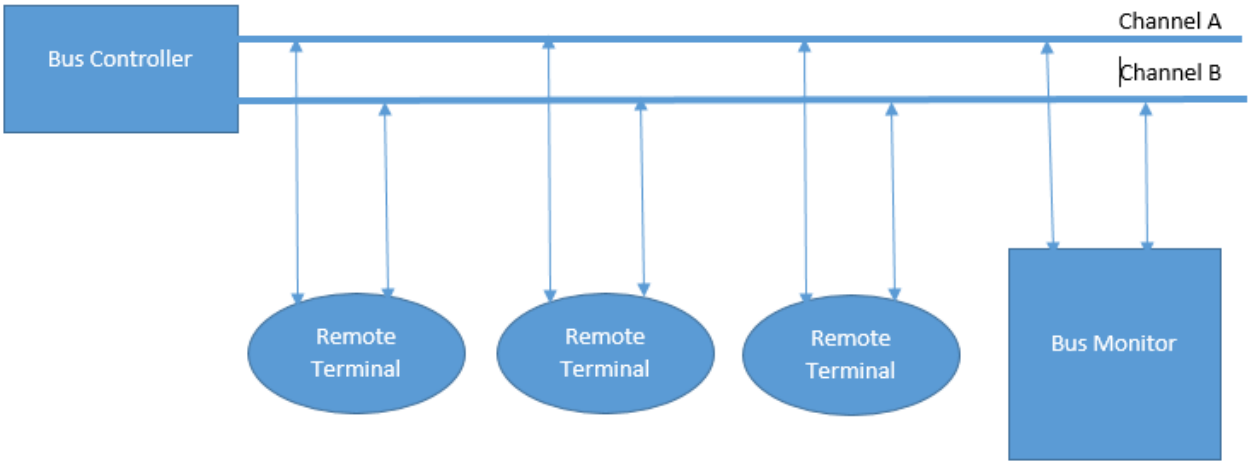

Figure 2-1: MIL-STD-1553 Architecture

Channel A and Channel B are two data paths in MIL-STD-1553 that make up the dual-redundant bus system. Channel A is the primary data path and is used for normal data transfer, while Channel B is the secondary data path and is used for redundant data transfer. The Channel B data is continuously monitored and compared with the data on Channel A, if a discrepancy is detected, the Channel A data is considered invalid and the Channel B data is used instead. The primary components of the MIL-STD-1553 architecture are defined below:

- A **Bus controller** is the device that manages the communication on the bus. It is responsible for initiating and controlling data transfer, as well as for managing the bus's timing and scheduling. The Bus controller also acts as the primary point of communication between the bus and the system's central computer. It can command and control all remote terminals connected to the bus.
- A **Bus Monitor** is a device that can monitor the bus traffic and record the data being transferred. It is typically used for diagnostic and troubleshooting purposes. The Bus Monitor can also be used to simulate a terminal on the bus for testing and development purposes.
- A **Remote Terminal** (RT) is a device that sends and receives data on the bus. This can be a sensor, actuator, or another type of device. A Remote Terminal (RT) is typically connected to one or more subsystems and interacts with the Bus controller to exchange information and commands with the central computer.

MIL-STD-1553 defines four main types of data transfer: BC-RT (Bus Controller to Remote Terminal), RT-RT (Remote Terminal to Remote Terminal), Modecode, and Broadcast.

- **BC to RT** data transfer works by the Bus Controller sending command messages to the Remote Terminal and the Remote Terminal responding with status messages. The command message contains a command word, specifying the action that the Remote Terminal should perform, and the status message contains a status word, indicating the outcome of the command. This process allows the Bus Controller to control and monitor the operation of the Remote Terminal.
- **RT to RT** data transfer is the process of transferring data between two Remote Terminals

with the help of the Bus Controller. One of the Remote Terminals sends a data message to the Bus Controller, which then forwards it to the intended Remote Terminal. The receiving Remote Terminal processes the information and sends a status message back to the Bus Controller, which then forwards it to the sending Remote Terminal. This process allows for greater control and monitoring of the data transfer process, as well as the ability to handle errors, collisions, and other issues that may occur during the data transfer.

- **Modecode** data transfer is a special type of data transfer used for initializing and configuring Remote Terminals. This type of transfer is initiated by the Bus Controller and is used to set the operating modes and parameters of the Remote Terminals.
- **Broadcast** data transfer is a type of data transfer where a single message is sent by the Bus Controller to all other devices on the bus. This type of data transfer is used for sending global commands or information to all devices on the bus.

2.2 ECP Overview

The ECP package is an R package that allows for nonparametric multiple change point analysis of multivariate data. It is designed to detect multiple change points in multivariate data, where each variable may have its change point. The package uses a nonparametric approach, meaning it does not make assumptions about the underlying distribution of the data. This allows for more flexibility in analyzing different types of data. It aims to overcome the limitations of existing change-point analysis by offering the ability to conduct multiple change-point analyses on both univariate and multivariate time series [3]. It can identify multiple change points without any prior knowledge of the number of changes [3].

The ECP package includes several different methods for detecting change points in multivariate data, such as Wild Binary Segmentation, Bayesian Online Changepoint Detection, PELT, and Segment Neighborhood. Among these methods, we are using the e-divisive method. This method is based on the idea of finding the point in a time series where the variance of the data changes significantly. It uses a recursive binary segmentation algorithm to search for change points in the data and is considered a non-parametric method that makes assumptions about the underlying distribution of the data, so it can be applied to different types of data.

Chapter 3 : Datasets and Feature Model

3.1 Datasets

The University of Victoria's Information Security and Object Technology (ISOT) Lab gathered the data utilized in this undertaking.

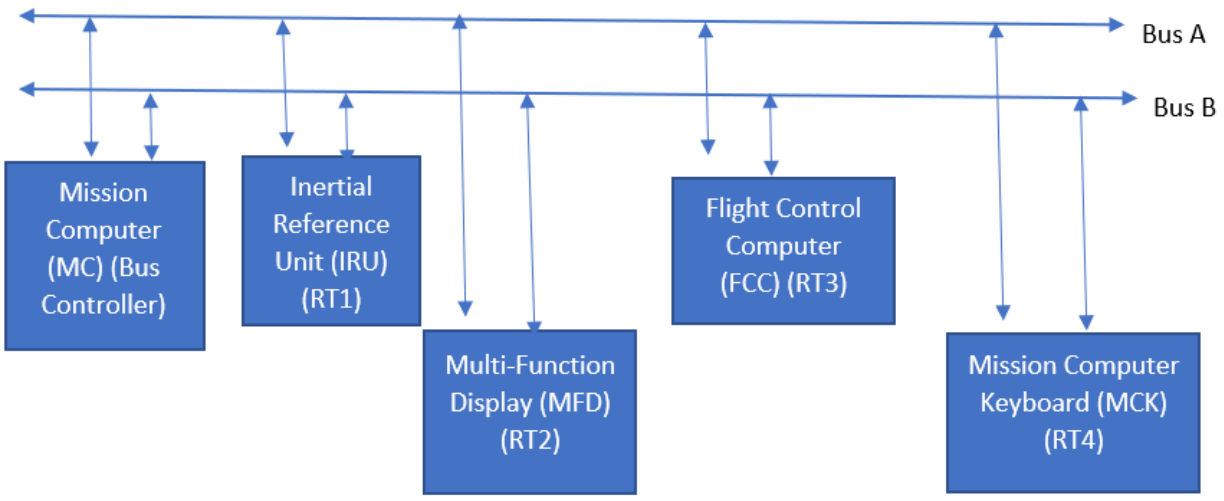

Figure 3-1: Baseline Architecture

The baseline topology for this project is illustrated in Figure 3.1. To generate data samples representative of normal activities, a core bus list was used. Additionally, by simulating various attack scenarios, attack samples were generated.

The baseline setup includes four remote terminals (RT1-RT4) and a bus controller. The remote terminals consist of an inertial reference unit (IRU), a multi-function display unit (MFD), a flight control computer (FCC), and a mission computer keyboard (MCK). The bus controller, known as the Mission Computer (MC), manages data transfers among the various components.

RT1, the inertial reference unit, provides navigation data to the MFD and FCC under the control

of the MC. RT2, the multi-function display unit, displays flight data from various components such as the MC, FCC, or IRU. RT3, the flight control computer, produces flight status information and receives navigation and flight path data from the IRU and the MC, respectively. Finally, RT4, the mission computer keyboard, receives input from the crew and transmits such data to the MC. The above setup was run for a duration of 10 minutes, resulting in the generation of 23,000 legitimate packets/messages. To facilitate data analysis, the obtained data was parsed and converted to CSV format. The format of the dataset is depicted in Table 3.1, which includes 55 fields.

Fields	Description
msgID	Sequence number associated with the message by the simulator
timestamp	Message timestamp
error	Indicate whether the error bit of the status word related to the message is set to contain TRUE/FALSE accordingly.
modeCode	Indicate whether or not the message is a mode command message that contains TRUE/FALSE accordingly.
channel	Channel associated with the message
connType	Communication type
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 at the receiving RT
wc	Word count: number of data words included in the message
modeCode value	Modecode value when applicable

txRsp	Transmit command response time
txSts	Transmit status word
rxRsp	Receive command response time
rxSts	Receive status word
dw0 … dw31	Values of the data word included in the message range from dw0 to dw31. N/A is used when there are no data words for a field.
Malicious	Indication of whether the message is malicious: contains TRUE/FALSE accordingly.
Injected	Indication of whether or not part of the data included in the message is injected: contains TRUE/FALSE accordingly.
gap	Inter-message gap time
msgTime	Message time

Table 3-1: Dataset Format

3.2 Attack Scenarios

In this section, we will discuss the attack vectors that were executed on the simulated MIL-STD-1553 architecture.

Following the generation of the benign dataset, attack samples were created by conducting a total of six different attacks, as listed below:

The first attack involved a Denial of Service (DOS) where a rogue terminal (RT0) repeatedly sent random words to the FCC (RT3) for 30 seconds, resulting in the creation of 148 packets/messages, some of which were malicious. The second attack was a Broadcast DOS, which again involved RT0 sending broadcast messages to the system. This attack was carried out for 30 seconds and resulted in the creation of 373 packets/messages, including both legitimate and malicious data.

In the third attack, RT0 replicated a message sent by a legitimate remote terminal (RT1-IRU) and

altered one of the data items (Altitude). This fake data injection attack resulted in 970 packets/messages being created over 30 seconds, including both legitimate and malicious data. The fourth attack was like the third, with the exception that the fake data was generated. This Noisy Injection attack was executed for 30 seconds and resulted in the creation of 970 packets/messages, including both malicious and legitimate data. The fifth attack was a logic attack, where a rogue remote terminal (RT) transmitted an unusual modecode value during an ongoing operation, which is not a normal request during flight. This attack resulted in the creation of 981 packets/messages over 30 seconds, some of which were malicious. Finally, the sixth attack was a combined attack, which incorporated elements of attacks four and five. This attack lasted 30 seconds and resulted in the creation of 1004 packets/messages, including both malicious and legitimate messages.

3.3 Feature Model

The MIL-STD-1553 standard defines a set of timing characteristics that are present in MIL-STD-1553 traffic. These properties include Data Throughput, Bus Utilization, Inter-Message Gap, Periodicity, and Response Time. Data Throughput is the maximum amount of data that can be transferred on the bus in each period and is crucial for determining the bus system's overall performance. Bus Utilization calculates the percentage of time that the bus is actively used for data transfer and ensures that the bus is being utilized efficiently. Inter-Message Gap determines the time duration between two consecutive messages transmitted over the bus and helps to prevent the bus from being overwhelmed with too many messages. Periodicity measures the regularity of messages sent on the bus and calculates the mean time duration between the Bus Controller transmitting messages to a specific Remote Terminal for a designated time frame. Finally, Response Time measures the time it takes for a Remote Terminal to respond to a command initiated by the Bus Controller.

In this study, a time frame of 250 ms was chosen because it yielded superior results. The features and techniques used to extract them are as follows:

- Data Throughput: The amount of data words transmitted during a 250 ms frame across the bus.
- Bus Utilization: The utilization rate of the bus for transmitting data during a 250 ms frame. It is measured in percentage.
- Inter-Message Gap: The duration elapsed from the moment the status word of a message is received by the BC to the moment when the command word of the subsequent message is transmitted.
- Modecode: The modecode commands sent by the Remote Terminal to the Bus Controller are extracted and compared with the standard operations. It is used to identify a particular kind of logic-based attack, where undesired signals are transmitted by a rogue Remote Terminal.

3.4 Intrusion Detection Approach

The ECP **e.divisive** algorithm is a statistical method for detecting multiple change points in a time series data. It works by recursively partitioning the data into smaller segments until no further significant change points can be found. Change point is a point in a time series where there is a significant change in the statistical properties, such as mean or variance. This can indicate a shift in the behavior of the system being monitored, which may be indicative of an intrusion or other anomalous event. The algorithm uses permutation tests to estimate the significance of the change points, and can handle a wide variety of data types, including continuous, discrete, and mixed data. The process of e-divisive algorithm is summarized below:

e-divisive algorithm

Input:

- X (dataframe of extracted features)
- sig_lvl (significance level)
- R (maximum distance threshold)
- k (number of clusters)
- min_size (minimum segment size)
- α (parameter)

Algorithm:

1. Compute pairwise distances between all points in X.
2. Test for significant change points using t-tests.
   (a) For each point i in X, calculate a t-statistic and p-value between the first i points and the remaining points.
   (b) If the absolute value of the t-statistic is greater than max_dist and the p-value is less than sig_lvl, add i to the list of change points.
3. If there are any change points, recursively apply the algorithm to each segment between change points.
   (a) For each pair of adjacent change points, call the $e\_divisive$ function on the segment between them.
   (b) Repeat Step 2 for each segment until there are no more change points.
4. Merge the segments to form the final output.
   (a) If k is not NULL, use k-means clustering to merge segments.
   (b) If k is NULL, merge segments based on min_size.

Output: List of indices representing the change points in X.

The e-divisive algorithm calculates the pairwise distance between all points in the input signal X to identify similarity between different time series data points. It then performs t-tests between adjacent segments to determine if there are significant changes in the statistical properties, such as mean or variance, between the segments. If a change point is identified, the algorithm recursively applies itself to each segment until there are no more change points. Finally, the segments are merged using either the k-means clustering algorithm, i.e., it groups segments with similar statistical properties into clusters or a minimum segment size, allowing for the identification of patterns within the data.

The algorithm for identifying attacks utilizing the ECP e-divisive method was implemented using the Python programming language and incorporates a well-known Python module for change detection.

The **e.divisive(X, sig.lvl=.05, R=199, k=NULL, min.size=30, alpha=1)** function is a part of the ECP package and it is used to perform the e-divisive method of change point detection on a multivariate time series data set **X**.

- **X** is a matrix or data frame with time-series data, where each column represents a different variable.
- **sig.lvl** is the significance level used for the hypothesis test to determine if there is a change point in the data. The default value is 0.05.
- **R** is the number of iterations to run the algorithm. The default value is 199.
- **k** is the maximum number of change points to detect. If **k** is not provided, the algorithm will detect as many change points as possible.
- **min.size** is the minimum size of a segment that can contain a change point. The default value is 30.
- **alpha** is the parameter of the exponential weighting function used to weight the p-value of the hypothesis test. The default value is 1.

The e.divisive algorithm takes as input a vector of observations X and uses a number of user-specified parameters to detect significant change points within the data. The sig.lvl parameter controls the significance level of the change point tests, while R determines the number of random permutations used in the tests. If the desired number of clusters is known, k can be set accordingly, otherwise, e.divisive will estimate the optimal number of clusters. The min.size parameter specifies the minimum number of observations in each resulting segment, and alpha serves as a

penalty factor for the number of change points detected in the final output. The output of e.divisive includes the following:

- **k_hat**: This is the estimated number of clusters within the data created by the change points detected by the algorithm.
- **order_found**: This is the order in which the change points were estimated by the algorithm. Each change point represents a significant shift in the underlying data generating process.
- **estimates**: These are the locations of the statistically significant change points detected by the algorithm.
- **considered_last**: This is the location of the last change point that was not found to be statistically significant at the given significance level. This can be interpreted as the last candidate change point that was tested by the algorithm.
- **p_values**: These are the approximate p-values estimated from each permutation test that was performed by the algorithm. Permutation tests are used to evaluate the statistical significance of the detected change points.
- **permutations**: This is the number of permutations performed by each of the sequential permutation tests used by the algorithm.
- **clusters**: This is the estimated cluster membership vector, which assigns each observation to a cluster based on its estimated change point.

Chapter 4 : Experimental Evaluation

4.1 Evaluation Procedure and Metrics

As the dataset comprises a collection of datasets specific to individual attacks, the experimental results of the algorithm for each attack are presented. By comparing the output against the available dataset, false positives and false negatives are identified. A detection graph is also plotted, which illustrates the changes detected for the timeline along with its amplitude for the output. Finally, the performance is calculated based on the following parameters, and Receiver Operating Characteristic (ROC) curves are plotted accordingly. The following evaluation metrics are computed:

- **Accuracy**: It is the proportion of the predicted change points to the overall data points.

$$Accuracy = \frac{True\ Positive + True\ Negative}{TP + TN + FP + FN}$$

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

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

- **True Positive Rate (TPR):** It is the proportion of the correctly identified positive change points to the total number of classified change points. It is also known as recall.

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

- **Mean Absolute Error (MAE):** It is the average of the absolute differences between the predicted and actual change points, divided by the total number of change points.

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

4.2 Experimental Results

For all the attack scenarios, we have used the values of e-divisive as sig.lvl=.05, R=199, min.size=2, and alpha=1 and the timeframe as 250ms respectively. We have modified the value of K as maximum, or none as needed to achieve the optimal results.

As demonstrated in Figures 4.1 and 4.2, the detection graph and ROC curve for the first Basic Denial of Service (DoS) attack are presented. The detection graph revealed the change points detected from the feature. The ROC curve shows the evaluated performance of the model. Here, the parameter for k was set to null as it gave the prominent result. As observed in Table 4.1, the feature of data throughput, bus utilization and combination of both performed better than the other features giving 100% accuracy.

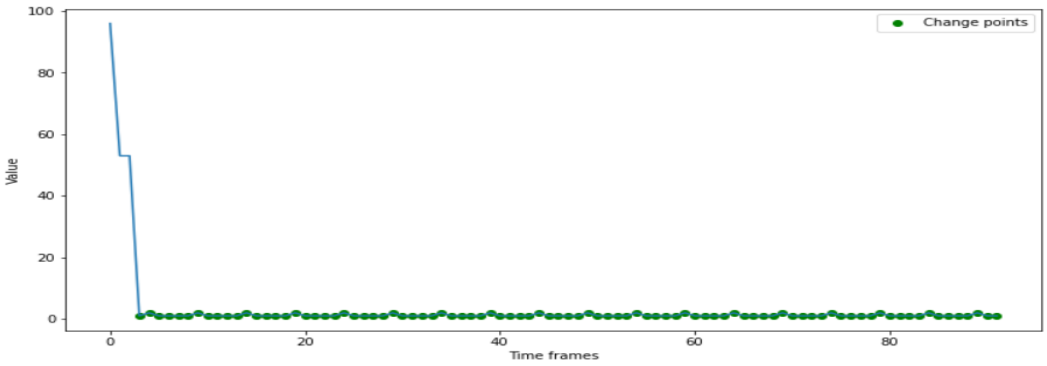

Figure 4-1: Change point detection graph: Attack 1 (Basic DOS) with Data Throughput feature

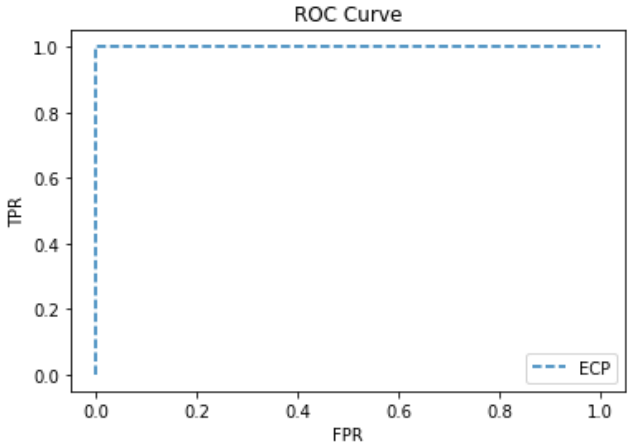

Figure 4-2: ROC Curve: Attack 1 with Data Throughput feature

Attack 1: DoS Attack (sig_lvl=.05, R=199, k=None, min_size=2, alpha=1)				
	Accuracy (%)	**FPR (%)**	**TPR (%)**	**MAE (%)**
Data Throughput	100.0	0.0	100.0	0.0
Bus Utilization	100.0	0.0	100.0	0.0
ModeCode	50.0	33.33	49.43	50.0
Inter Message gap	52.08	27.27	39.32	47.91
Data Throughput + Bus Utilization	100.0	0.0	100.0	0.0
ModeCode + Inter Message gap	43.0	0.0	37.07	57.0

Table 4-1: Results: Attack 1

As illustrated in Figures 4.3 and 4.4, the detection graph showing the change points detected in the timeframe of data throughput feature and ROC curve shows trade-off between the TPR and FPR. for Attack 2 (Broadcast Denial of Service (DoS)) are presented. Like the previous attack the parameter for k was set to null. Table 4.2 indicates that the data throughput feature is more effective than the other features in this attack scenario showing 100 % accuracy with 0% FPR rate.

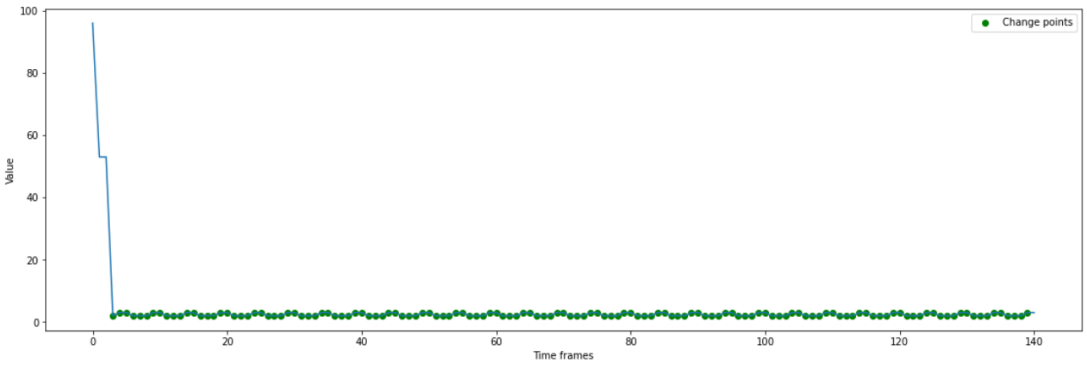

Figure 4-3: Change point detection graph: Attack 2 (Broadcast DOS) with Data Throughput feature

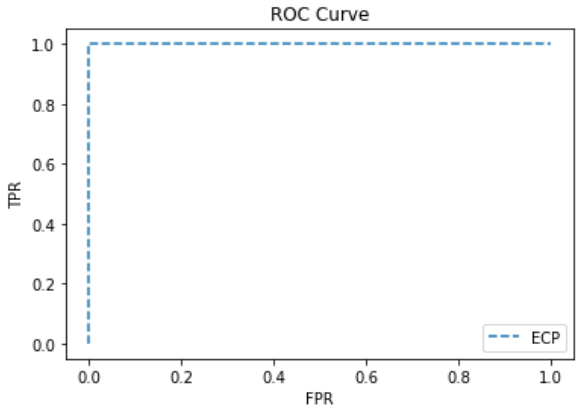

Figure 4-4: ROC Curve: Attack 2 with Data Throughput feature

Attack 2: Broadcast DoS Attack (sig_lvl=.05, R=199, k=None, min_size=2, alpha=1)				
	Accuracy (%)	**FPR (%)**	**TPR (%)**	**MAE (%)**
Data Throughput	100.0	0.0	100.0	0.0
Bus Utilization	50.0	33.33	49.71	50.0
ModeCode	50.47	25.0	50.0	49.52
Inter Message gap	55.82	32.90	36.95	44.17
Data Throughput + Bus Utilization	41.11	0.0	39.85	58.86
ModeCode + Inter Message gap	43.0	0.0	37.07	57.0

Table 4-2: Results: Attack 2

As can be seen in Table 4.3, the inter-message gap feature outperforms all other features in Attack 3 (Subtle Injection) giving 97.33 % accuracy and 2.66% MAE. The value for k was set to maximum for this attack as it generated best results. Figures 4.5 and 4.6 present the detection graph presenting the change points detected from inter-message gap feature for subtle injection attack and ROC curve depicts ratio between TPR and FPR, respectively.

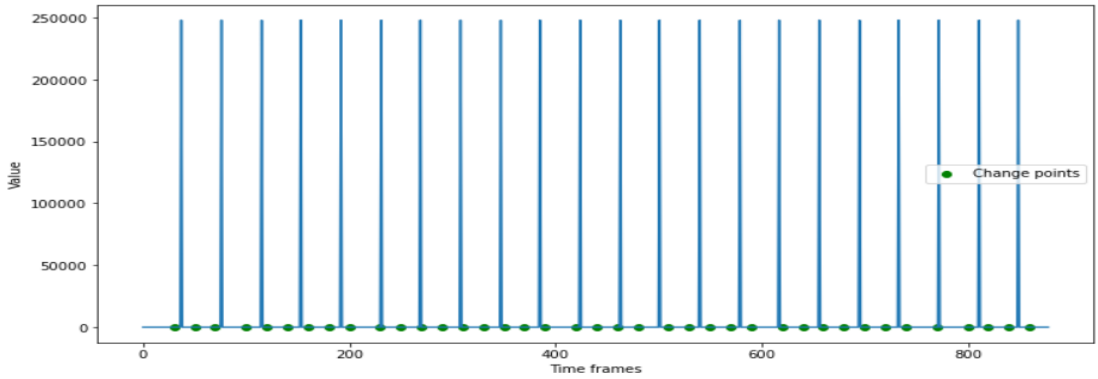

Figure 4-5: Change point detection graph: Attack 3 (Subtle injection) with Inter-message gap feature

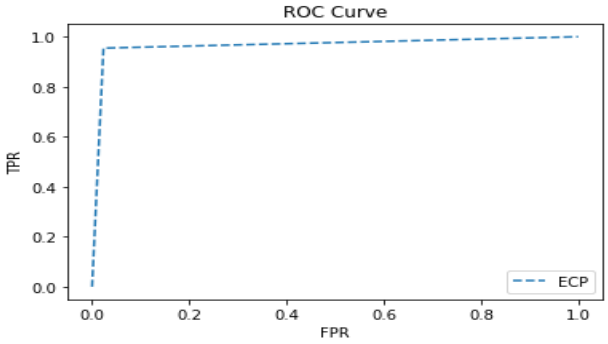

Figure 4-6: ROC Curve: Attack 3 with Inter Message gap feature.

Attack 3: Subtle Injection Attack (sig_lvl=.05, R=199, k=max, min_size=2, alpha=1)				
	Accuracy (%)	**FPR (%)**	**TPR (%)**	**MAE (%)**
Data Throughput	79.64	41.17	45.45	43.26
Bus Utilization	58.86	48.35	36.36	41.13
ModeCode	91.82	6.96	45.45	8.17
Inter Message gap	97.33	2.34	95.45	2.66
Data Throughput + Bus Utilization	56.63	39.56	40.90	43.36
ModeCode + Inter Message gap	89.66	8.30	36.36	10.33

Table 4-3: Results: Attack 3

Figures 4.7 and 4.8 present the detection graph which displays the detected change point from the timeframe of the feature and ROC curve demonstrates relationship between TPR and FPR for Attack 4 (Noisy Injection), respectively. As indicated in Table 4.4, the inter-message gap feature generating 96 % accuracy is more effective than the other features in this scenario.

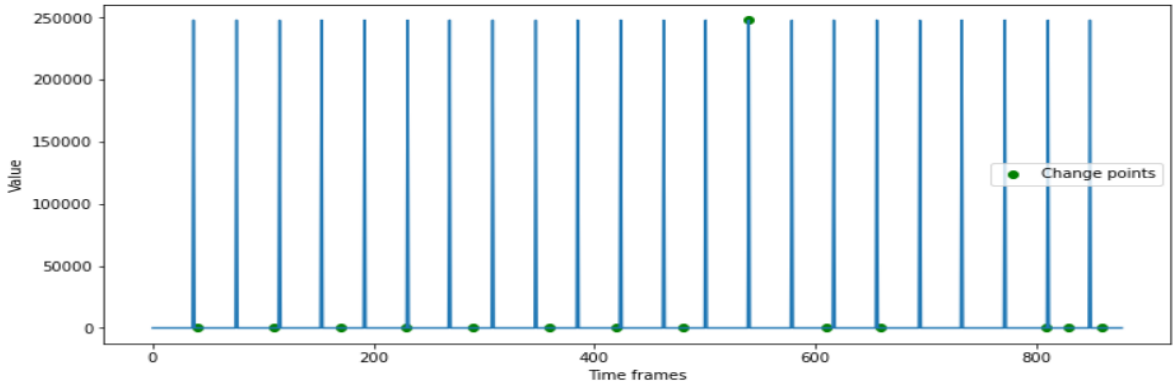

Figure 4-7: Change point detection graph: Attack 4 (Noisy injection) with Inter-message gap feature

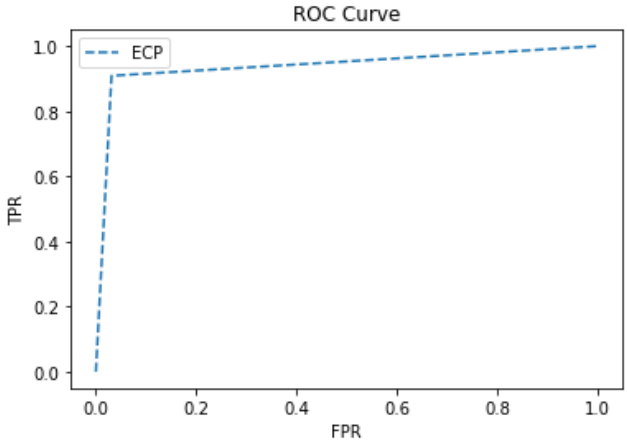

Figure 4-8: ROC Curve: Attack 4 with Inter Message gap feature

Attack 4: Noisy Injection Attack (sig_lvl=.05, R=199, k=max, min_size=2, alpha=1)				
	Accuracy (%)	FPR (%)	TPR (%)	MAE (%)
Data Throughput	65.24	36.13	72.72	34.74
Bus Utilization	67.37	33.61	72.72	32.62
ModeCode	49.64	49.7	45.45	50.35
Inter Message gap	96.0	3.12	90.90	4.0
Data Throughput + Bus Utilization	67.37	33.61	72.72	32.62
ModeCode + Inter Message gap	54.60	41.17	31.81	45.39

Table 4-4: Results: Attack 4

As displayed in Table 4.5, the ModeCode feature outperforms all other features giving accuracy of 97.91 % and MAE 2.08% for Attack 5 (Logic attack). Figures 4.9 and 4.10 present the detection graph which reveals the change points detected from using the modecode feature and ROC curve demonstrates relationship between TPR and FPR for the attack. The parameter k was set to maximum for detecting all the change points.

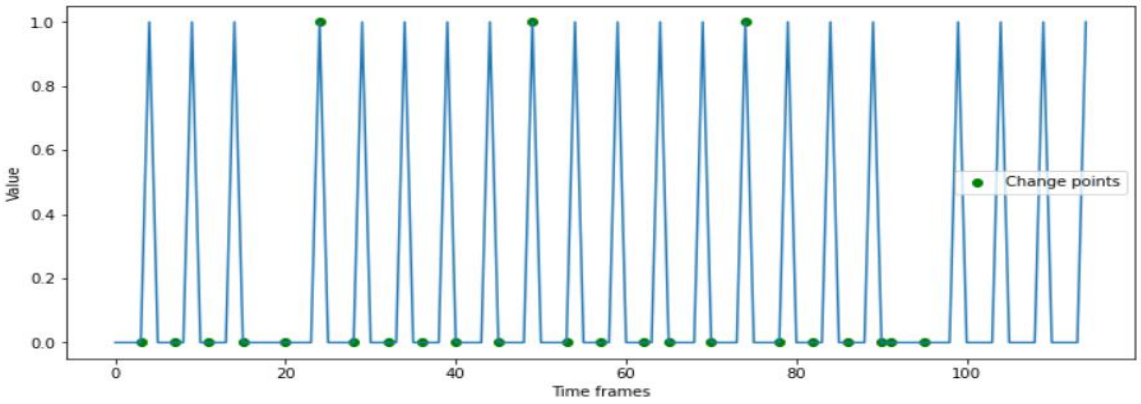

Figure 4-9: Change point detection graph: Attack 5 (Logic) with Modecode feature

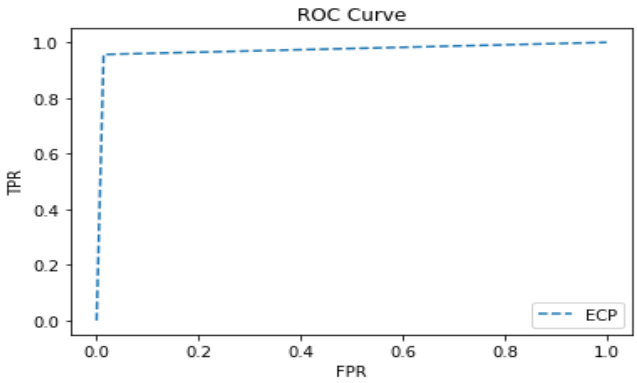

Figure 4-10: ROC Curve: Attack 5 with ModeCode feature

Attack 5: Logic Attack (sig_lvl=.05, R=199, k=max, min_size=2, alpha=1)				
	Accuracy (%)	**FPR (%)**	**TPR (%)**	**MAE (%)**
Data Throughput	74.78	28.26	86.95	25.21
Bus Utilization	58.04	39.16	43.47	41.95
ModeCode	97.91	1.36	95.65	2.08
Inter Message gap	57.69	41.66	55.0	42.30
Data Throughput + Bus Utilization	62.60	35.86	56.52	37.39
ModeCode + Inter Message gap	58.26	38.04	43.47	41.73

Table 4-5: Results: Attack 5

Figures 4.11 and 4.12 present the detection graph showing the detected change points obtained from the inter-message gap feature and ROC curve showing trade-off between TPR and FPR for Attack 6 (Combination attack), respectively. As outlined in Table 4.6, the inter-message gap feature is more effective than the other features in this attack scenario by generating 99.12% accuracy and 0% FPR.

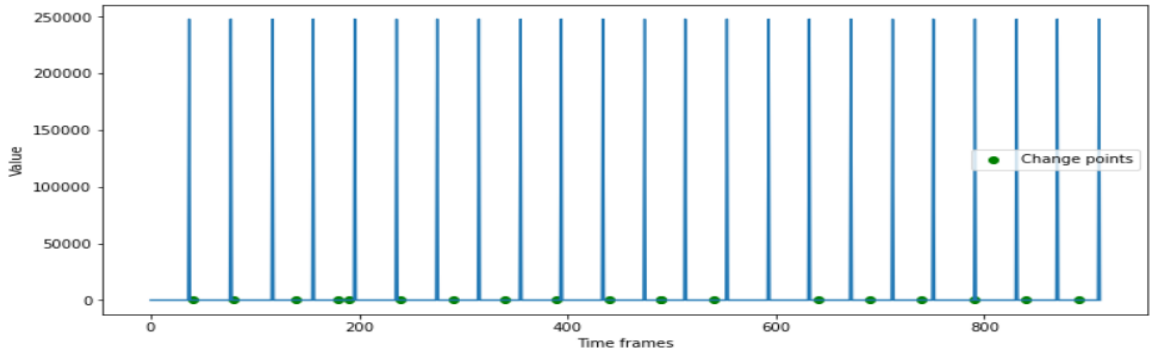

Figure 4-11: Change point detection graph: Attack 6 (Combination) with Inter-message gap feature

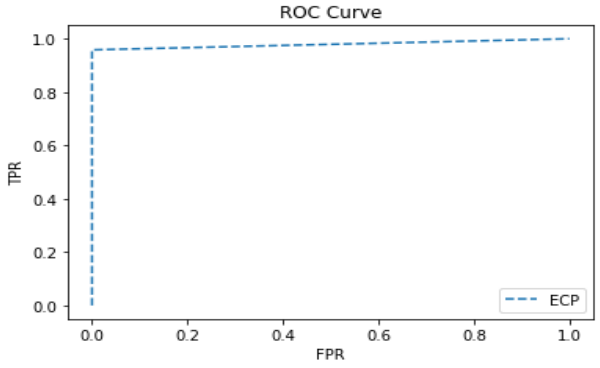

Figure 4-12: Attack 6 with Inter Message gap feature

Attack 6: Combination Attack (sig_lvl=.05, R=199, k=max, min_size=2, alpha=1)				
	Accuracy (%)	**FPR (%)**	**TPR (%)**	**MAE (%)**
Data Throughput	77.39	25.55	88.0	22.60
Bus Utilization	63.47	35.55	60.0	36.52
ModeCode	51.30	47.77	48.0	48.69
Inter Message gap	99.12	0.0	95.83	0.8
Data Throughput + Bus Utilization	72.17	30.0	80.0	27.82
ModeCode + Inter Message gap	57.29	38.35	43.47	42.70

Table 4-6: Results: Attack 6

4.3 Results Discussion

The performance evaluation of MIL-STD-1553 bus systems using the ECP e-divisive algorithm showed that the data throughput and bus utilization features gave the best results for detecting DOS attacks. This indicates that these features are effective in detecting high-volume data transfer that can be indicative of a DOS attack. In addition, the data throughput feature was found to be the best suited for detecting Broadcast DOS attacks, which occur when an attacker sends high-volume data to all Remote Terminals on the bus. The e-divisive algorithm can detect this attack by analyzing the data throughput feature. The inter-message gap feature gave the best result for detecting Subtle injection and noisy injection attacks. These attacks involve an attacker injecting false data into the system in a way that appears legitimate. The e-divisive algorithm can detect these attacks by analyzing the inter-message gap between valid data and identifying any irregularities in the gap. The Modecode feature was found to be best suited for detecting Logic attacks, which involve an attacker manipulating the control commands sent to a Remote Terminal to cause it to behave in unexpected ways. The e-divisive algorithm can detect this attack by analyzing the Modecode and identifying any unusual patterns or commands.

Finally, the e-divisive algorithm was found to accurately detect a combination of Logic and fake injection attacks using the inter-message gap feature.

Chapter 5 : Conclusion

MIL-STD-1553 is a reliable and robust communication protocol that ensures the efficient transfer of data on the bus. However, the increasing use of MIL-STD-1553 in critical systems has led to a growing concern about the security of these systems. Intrusion detection is an important aspect of securing these systems and preventing unauthorized access.

The evaluation of MIL-STD-1553 bus systems using the ECP e-divisive algorithm demonstrated its effectiveness in detecting various types of attacks. Four features were extracted to detect the change points in the data, which led to detection of six different types of attacks. After conducting extensive performance evaluations and tests, it was demonstrated that the results indicated that individual features were successful in detecting specific types of intrusions.

For future work, several possibilities exist. One approach is to investigate the use of different properties to detect change points in the data. Additionally, combination of features can also be utilized to improve the accuracy and robustness of intrusion detection techniques. Finally, the output of the e-divisive algorithm can be used as input to a logic-based system to develop a more robust defense strategy against potential attacks.

Moreover, further research can be done into the resilience of the e-divisive algorithm to adversarial attacks, as attackers may attempt to manipulate the data to evade detection. Finally, the scalability of the algorithm should be investigated, particularly in large-scale systems where the volume of data can be overwhelming. These directions could help to improve the accuracy, efficiency, and robustness of intrusion detection techniques for MIL-STD-1553 bus systems.

References

[1] R. M. McGraw, M. J. Fowler, D. Umphress, and R. A. MacDonald, "Cyber threat impact assessment and analysis for space vehicle architectures," in *SPIE Defense+ Security, International Society for Optics and Photonics*, United States, 2014.

[2] 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.

[3] N. A. James and D. S. Matteson, "ecp: An R Package for Nonparametric Multiple Change Point Analysis of Multivariate Data," *Journal of Statistical Software*, vol. 62, Dec 2014.

[4] B. Losier, R. Smith, and V. Roberge, "Design of a time-based intrusion detection algorithm for the MIL-STD-1553," Royal Military College of Canada, Kingston, ON, Rep. 2102, 2019.

[5] H. Saad, I. Traore, P. Quinan, K. Ganame and O. Boudar, " A Collection of Datasets for Intrusion Detection in MIL-STD-1553 Platforms," in *Artificial Intelligence for Cyber-Physical Systems Hardening*, I. Traoré, I. Woungang, and S. Saad, Eds. Springer, 2022, Chapter 4.

[6] A. Aldribi, I. Traoré, B. Moa, and O. Nwamuo, "Hypervisor-based cloud intrusion detection through online multivariate statistical change tracking," *Computers & Security*, vol. 88, Jan 2019.

[7] R. Yahalom, D. Barishev, A. Steren, Y. Nameri, M. Roytman, A. Porgador, and Y. Elovici, "Datasets of RT spoofing attacks on MIL-STD-1553," *Data in brief*, vol. 23, April 2019.