Summary Report: Security Analysis of the GOOSE Communication Protocol Using Statistical Profiling

Student Name: **Simon Wachiuri**

Home University: **Strathmore University (SU)**

Home Faculty: **Faculty of Information Technology (FIT)**

Academic Level: **Masters Student**

Field of Study: **Computer Science**

Host University: **Brno University of Technology (BUT)**

Host Faculty: **Faculty of Information Technology (FIT)**

Supervisor: **Ing. Petr Matoušek, Ph.D., MA**

1. Project Overview

The IEC 61850 international standard for substation and power systems communication defines a common protocol that implements abstract models of primary and secondary substation equipment, communication systems, functional characteristics, structure of data packages in the Intelligent Electronic Devices (IEDs) and the relationship between them. The protocol constitutes the following parts: Static Configuration description Language (SCL) - an XML based definition of the description of the parts of a substation, Communication profile (IEC 61850 stack) which includes a number of possible communication profiles and service mappings, Communication Services which implement the facilities of communication between servers and clients, Logical node data models and conformance tests.

In the wake of the emerging industrial innovation, particularly for the communication networks and protocols, securing the implementation of the mapping protocols of the IEC 61850 for resilience of performance of the electrical substation systems is highly significant. The current mappings in the IEC 61850 standard include the Manufacturing Message Specification (MMS), Generic Object-Oriented Substation Event (GOOSE) and the Sampled Measured Values (SMVs). This study centres on the GOOSE protocol mapping in its classical model of substation Local Area Networks (LANs) and its integration to the Transmission Control Protocol/Internet Protocol (TCP/IP).

The study focuses on the cybersecurity implementation that can extract anomalies in the operation of the GOOSE messaging approach. The peer-to-peer communications in IEC 61850 integrated substation protection and control system are based on what is defined as GOOSE messages. These communications use multicast Ethernet communications and represent the asynchronous reporting of the IEDs functional state based on the message exchange. GOOSE messages replace the hard-wired control signals exchanged between IEDs for status switching. Notably, GOOSE messages are not command-drivers and therefore do not tell any receiving IEDs what to do. They just indicate that a new event has occurred, what that even is and the time when it happened.

The practical demonstration of this study, therefore, implements a statistical fingerprint on the GOOSE message to illustrate a scenario that identifies a correct (non-anomalous) GOOSE message from an incorrect (possibly compromised) GOOSE message. The study implements a statistical algorithm that mimics a supervised learning approach based on a training dataset and a testing dataset. Comparatively, the datasets are tested to distinguish the datasets that have a known traffic flow (correct GOOSE message) from the ones whose traffic flow is unknown or experienced an attack (incorrect GOOSE message).

Keywords: IEC 61850, GOOSE messages, Anomaly Detection, Cyber Security, Statistical classification.

2. Literature Studies

The following summary of studies present the source of literature inspiring this work:

1. Traffic classification through simple Statistical Fingerprinting – Manuel Crotti, Maurizio Dusi, Franceso Gringoli, Luca Salgarelli

The authors of this work (Crotti, Gringoli, & Salgarelli, 2007) presents a flow classification mechanism based on three simple properties of the captured IP packets: Size, Interarrival time and Arrival order. Although these quantities have already been used in the past, the paper contribution is based on new structures of protocol fingerprints and a simple classification algorithm based on normalised thresholds. This approach tries to classify network traffic relying exclusively on the statistical properties of flows.

The rationale behind the use of packet size, inter-arrival times and arrival order (of packets) for the classification of network flows lies in the observation that at least during the beginning stage of each layer-4 connection, the statistics related to each of these quantities depend mostly on the application-layer state machine that has generated the flow. Examples providing this truth are HTTP data request, the authentication stage of POP3 retrieval and SMTP helo_sender_receiver agreements. The data flow from the exchange of TCP segments resulting from two applications talking to each other should be broken into packets and time such packets in a way that is very specific to protocol-dependent statistics.

Another related study carried out by (Karagiannis , Papagiannaki, & Faloutsos, 2005) demonstrate a method based on the analysis of host behaviour whose goals introduce a classification of flows according to the applications that generate them without payload analysis. The approach has the same goals on classification as (Crotti, Gringoli, & Salgarelli, 2007), however it differs considerably in their method of classification which is implemented by associating a host behaviour pattern to one or more applications and then refining the association by means of heuristics and behaviour stratification.

2. Security Monitoring of IoT Communication Using Flows – Petr Matoušek, Ondřej Ryšavý, Matěj Grégr

In this literature, (Matoušek, Ondřej , & Matěj , 2019) demonstrate an IoT Flow monitoring using an IoT enabled IPFIX probe that monitors IoT traffic, parses headers of IoT protocols and extracts metadata from the headers. This is useful for IoT network monitoring that observes packets, extracts selected data from IoT headers and maps them into IoT enabled IPFIX records.

Figure 1 shows their implementation of an extended IPFIX record enriched by CoAP header field values. This example, which can be used for IoT monitoring using an extended IPFIX record for CoAP monitoring, shows three CoAP enabled IPFIX records extended by CoAP message type (GET, Content or PUT), message ID, token ID (TKN), and resource identifier in the form of URI. For each IoT protocol, we define a specific set of headers useful for monitoring and in this case of CoAP monitoring IoT packets having the same key properties, create one IoT extended IPFIX record that represents these packets.

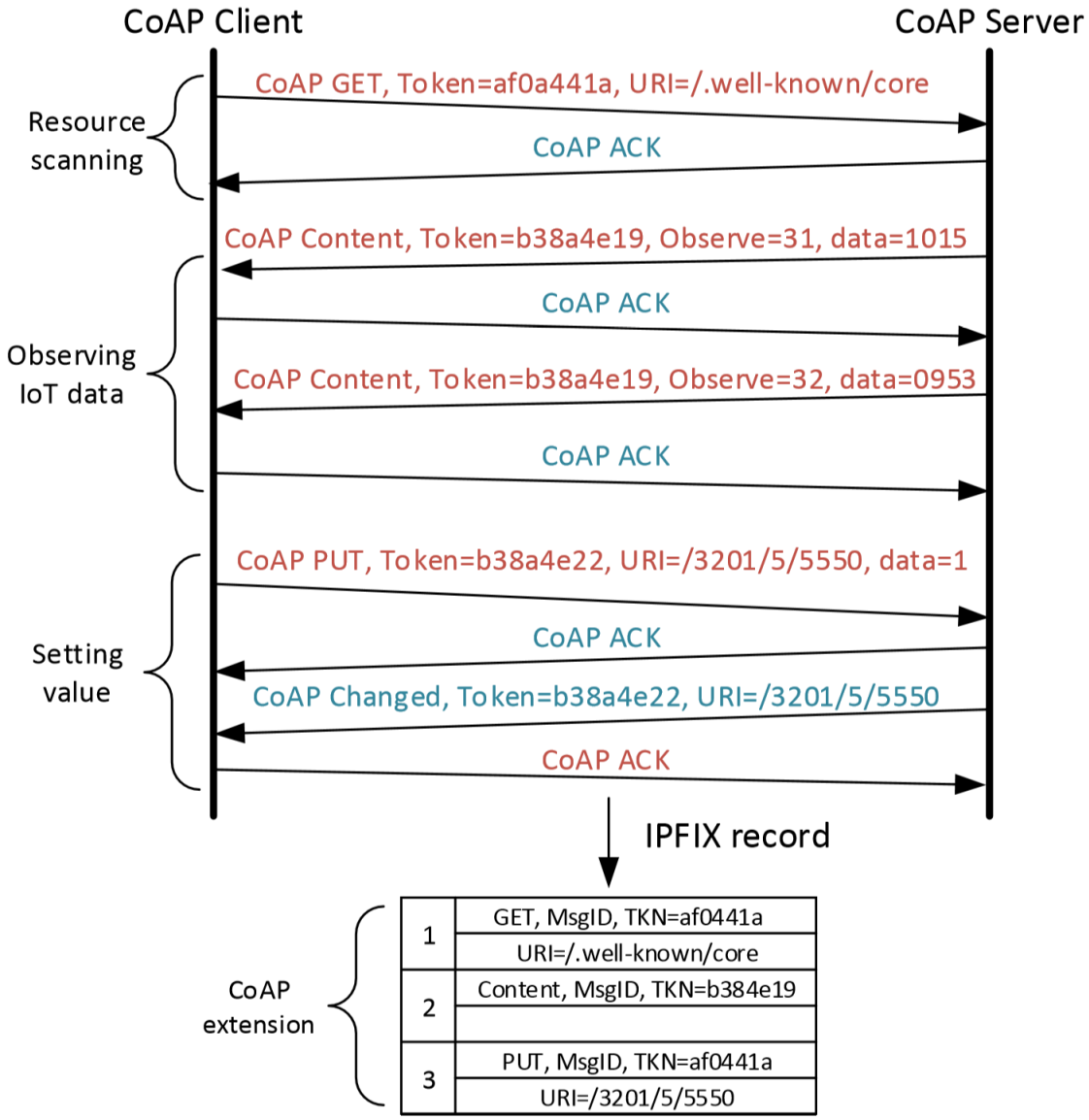

Figure 1: Extended IPFIX Record for CoAP Monitoring

The IoT traffic examined by (Matoušek, Ondřej , & Matěj , 2019) assumes regular traffic to have unique features such as the number of packets sent, timing and amount of data exchanged. A statistical model for CoAP operations is created by these authors and a model demonstrated using two input paraments which take on: the number of messages and the amount of data within a fixed period.

The developed model reliably describe normal behavior and detect significant deviations from a probability distribution standpoint. The model can be modified to take on other statistical properties such as maximum and minimum packet size and inter-packet delay distribution among other parameters. The method, albeit, cannot be applied to encrypted communication provided by DTLS since it relies on the possibility of inspecting CoAP header fields.

The modeling of the proposed method learns patterns of CoAP resource usage and creates a systems wide profile. Typically, the client operation r^{op} involves sending a resource operation usage to server with its resource address r^{uri}. The statistical information Model *M*, that relates

to an operation r^{op} on the resource r^{uri} is characterised by: Variables $X_1 \ldots X_k$ where X_1 is no. of packets and X_2 is no. of octects associated with r^{op} and Usage of monitored resource within specific period of observation i.e. time windows.

The steps adopted in the creation of the learning profile here reflect implementations carried out by (Crotti, Gringoli, & Salgarelli, 2007) as well. A collection of resource usage models called a usage profile P, is created during the learning process and follows the below steps:

i. During a fixed interval regular network communication is captured and it's considered in a set of time windows ($w_1 \ldots, w_t$),
ii. CoAP flows grouped for resource usage label r= (r^{op},r^{uri}) in each Time window w_i, example an operation of node/IP address with URI resource of floor_1_light,
iii. Aggregated set of flows {$e_1 \ldots, e_m$} belonging to the same resource usage r, a set of feature vectors extracted for example a light sensor identified by UriPath /floor_1_light and running on host 192.168.10.107 received 4.020 CoAP packets with 179.879 bytes within the given period.
iv. Set of samples is fed into an EM algorithm with the model given as a joint probability function and a computed threshold value.

The discrimination of the CoAP traffic, as consequently shown by (Matoušek, Ondřej , & Matěj , 2019), is the calculation of the probability of the observed behaviour which could be normal or abnormal.

3. A Review of Research Work on Network-based SCADA Intrusion Detection System (IDS) by Slavica V, Boštjančič Rakas, Mirjana D. Stojanović, Jasna D. Marković-Petrović

The study presented by (Slavica , Rakas, & Stojanović, 2020) assesses the state-of-the-art, identifies the open issues and provides an insight for future study areas on how specific intrusion detection systems (IDSs) are needed to secure modern supervisory control and data acquisition (SCADA) systems.

The authors begin their journal with an analysis of the factors that affect the design of dedicated intrusion detection systems in SCADA networks and focus on network-based IDS solutions. They propose a structured evaluation methodology that encompasses detection techniques, protected protocols, implementation tools, test environments and IDS performance. Moreover, they provide a brief description and evaluation of 26 selected research papers, published in the period 2015–2019.

Under the study I present in this paper, my focus, similarly delves into the classification of intrusions detection methodologies in SCADA systems, detection accuracy and a brief description of selected models for the Goose Protocol and Statistical anomaly. The key points drawn from this study are described herein.

(Slavica , Rakas, & Stojanović, 2020) , explain various terminologies used in regards to classification of intrusion detection methodologies in SCADA systems. The most general classification of intrusion detection methodologies is the blacklist and whitelist approaches. Blacklist approaches assume that all processes/requests are approved unless they are explicitly mentioned on the blacklist. Whitelist approaches profile "normal behavior" so that deviations can be reported. Their paper elucidate that basic detection methodologies comprise signature-based detection, anomaly-based detection and specification-based detection.

Anomaly-based detection is a whitelist approach, which includes techniques that compare monitored events with the list of activities, which were predefined as normal to identify significant deviations. The general advantage of anomaly-based techniques refers to efficient detection of unknown threats. The generic functional architecture of anomaly-based IDS is depicted in Figure 2 (a). In the preprocessing phase, the observed instances are represented in a predefined form. IDS creates static or dynamic models (profiles) representing normal behavior of users, hosts, network connections, and applications. During a training period the initial profile is generated which can be done in different ways, depending on the IDS type. According to the nature of processing involved in the behavioural model, anomaly-based techniques can be classified into three main categories: statistical-based, knowledge-based, and machine learning-based, as illustrated in Figure 2(b).

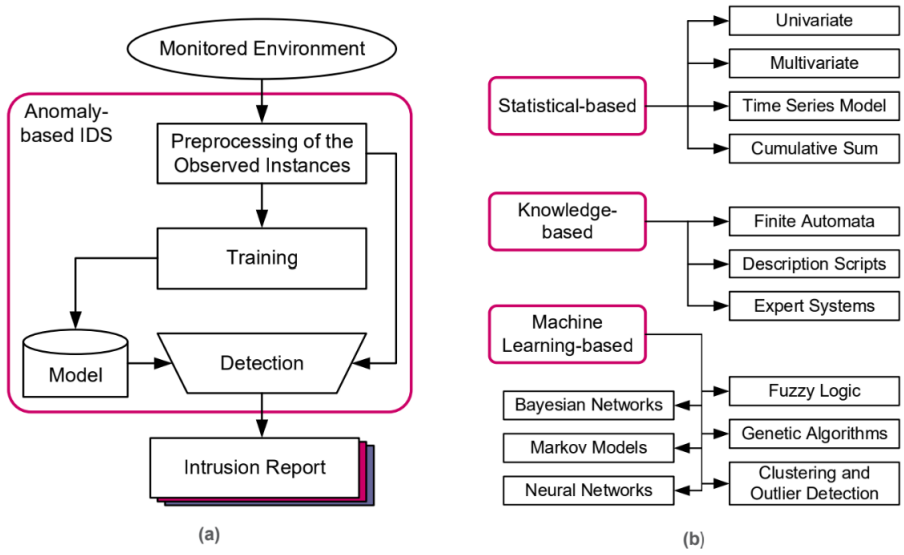

Figure 2: Anomaly-based IDS: (a) functional architecture and (b) classification tree

Relatedly, my study narrows down to the Statistical-based techniques using statistical properties and tests to determine whether the observed behavior deviates significantly from the expected behavior. They include a number of techniques based on univariate, multivariate, time-series models and cumulative sums (CUSUM). The advantages of statistical-based techniques include the ability to learn the expected behavior of the system (without prior knowledge about its normal activity) and the ability to provide accurate long-term detection of malicious activities. Their main disadvantage refers to possibility that the attacker can train the system in such a way that the malicious traffic is considered as normal.

Detection accuracy (also known as classification accuracy or effectiveness), as presented by (Slavica , Rakas, & Stojanović, 2020) represents the ability of the system to distinguish between intrusive and non-intrusive activities. It is represented by a set of measures that determine how correctly an IDS works. Confusion matrix represents true and false classification results, as indicated in figure 3.

Confusion matrix	Derived evaluation metrics
<table><tr><td rowspan="2">Actual</td><td colspan="2">Predicted</td></tr><tr><td>Attack</td><td>Normal</td></tr><tr><td>Attack</td><td>*TP*</td><td>*FN*</td></tr><tr><td>Normal</td><td>*FP*</td><td>*TN*</td></tr></table>	1. $FPR = FP/(FP + TN)$ 2. $FNR = FN/(TP + FN))$ 3. $DR = TPR = Recall = TP/(TP + FN) = 1 - FNR$ 4. $TNR = TN/(FP + TN) = 1 - FPR$ 5. $Accuracy = (TP + TN)/(TP + TN + FP + FN)$ 6. $Precision = TP/(TP + FP)$ 7. $F\text{-}measure = 2/(1/Precision + 1/Recall) = 2TP/(2TP + FP + FN)$

Figure 3: Confusion matrix and derived evaluation metrics.

The variables of confusion matrix are:

- True positive (TP) – number of successfully detected malicious activities;
- True negative (TN) – number of normal activities that are successfully labeled as non-intrusive;
- False negative (FN) – number of malicious activities that are not detected, but considered as normal;
- False positive (FP) or false alarm (FA) – number of normal activities that are detected as malicious

In Figure 3, the authors present different evaluation metrics that are derived as functions of the confusion matrix variables. Those metrics are as follows:

1. False positive rate (FPR) measures the ratio between the number of normal instances detected as attacks and the total number of normal activities.
2. False negative rate (FNR) measures the ratio between number of malicious activities that are not detected and the total number of malicious activities.
3. Detection rate (DR), also known as True Positive Rate (TPR) or Recall, measures the fraction of anomalies that are successfully identified.
4. True Negative Rate (TNR) measures the ratio between the number of normal instances detected as non-intrusive and the total number of normal activities.
5. Accuracy measures the fraction of instances that are correctly classified.
6. Precision denotes the probability that a detected anomaly is correct.
7. F-measure represents the weighted harmonic mean of Precision and Recall.

4. A Behaviour-based Intrusion Detection Technique for Smart Grid Infrastructure by *Y.* J. Kwon, H. K. Kim, Y. H. Lim, and J. I. Lim

The authors, (Kwon , Kim, Lim, & Lim, 2015) in a related paper, present a GOOSE implementation of a keen interest to my study. They focus on GOOSE Protocol and statistical anomaly techniques. They propose an intrusion detection technique for IEC 61850 substations, which focuses on GOOSE and MMS protocols, taking into account specification-based metrics and multivariate analysis of network features. To detect malicious traffic, their proposed technique uses static and dynamic features. The static features verifies the syntax correctness of the protocol. Dynamic features depend on the network environment. Their anomaly detection represents a function of the three weighted input parameters, i.e., network metric, GOOSE metric and MMS metric.

According to (Kwon , Kim, Lim, & Lim, 2015), a static feature is used to check the consistency or grammatical correctness of the protocol. In this case, they use a response and report feature in a measurement signal of MMS protocol-based command as a static feature. A dynamic feature, on the other hand, is a variable related to a network environment. For instance, GOOSE protocol is changeable on the frequency and the distribution of GOOSE message according to each implementation of real substation site. To monitor GOOSE usage pattern efficiently, they propose a Recency-Frequency-Monetary (RFM) analysis to capture GOOSE behavior-based pattern as a dynamic feature. In addition, generic traffic features (e.g. bits per second (bps), packets per second (pps), and connections per second (cps)) are used as dynamic features to increase overall detection accuracy.

(Kwon , Kim, Lim, & Lim, 2015) define an anomaly detection function whose design is defined as function A of three input parameters A (f(network_metric), f(GOOSE _metric), f(MMS_metric)}. It can be represented by the total anomaly possibility value:

A = {wI·f(bps, pps, cps) + w2·f(GOOSE RFM) + w3·f(MMS command)} (1)

where WI. W2, W3 represent weight values from ° to 1 given, satisfying the sum of three weights equals 1. As an initial weight value, WI. W2, W3 are selected as 0.4, 0.4 and 0.2 in order since dynamic features can sensitively explain real-time network traffic data.

3. Implementation Architecture

Experiments (GOOSE Communication)

The following gics-goose.pcap file is a sample GOOSE communication and with the summary of some of the features like number of packets, size of pcap, duration and others shown in Figure 4 below.

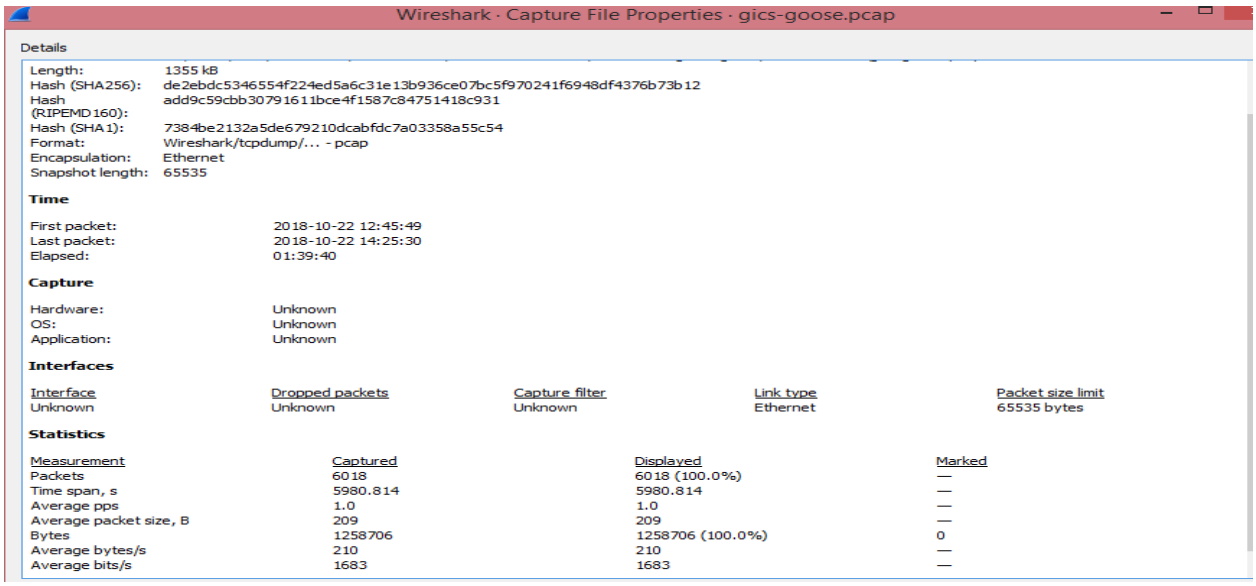

Figure 4: GOOSE pcap file features

The tools considered for exploring interesting features of the goose communication were Wireshark, Tshark and OMICRON IEDScout. For the purposes of this scope of this experiment, wireshark and tshark will be used to confirm that the features of GOOSE datagram and any payload it may contain. This is because no simulation will be performed at this stage. As can be seen in Figure 5 below, goose is encapsulated into an Ethernet frame and can be confirmed by type: IEC 61850/GOOSE (0x88b8)

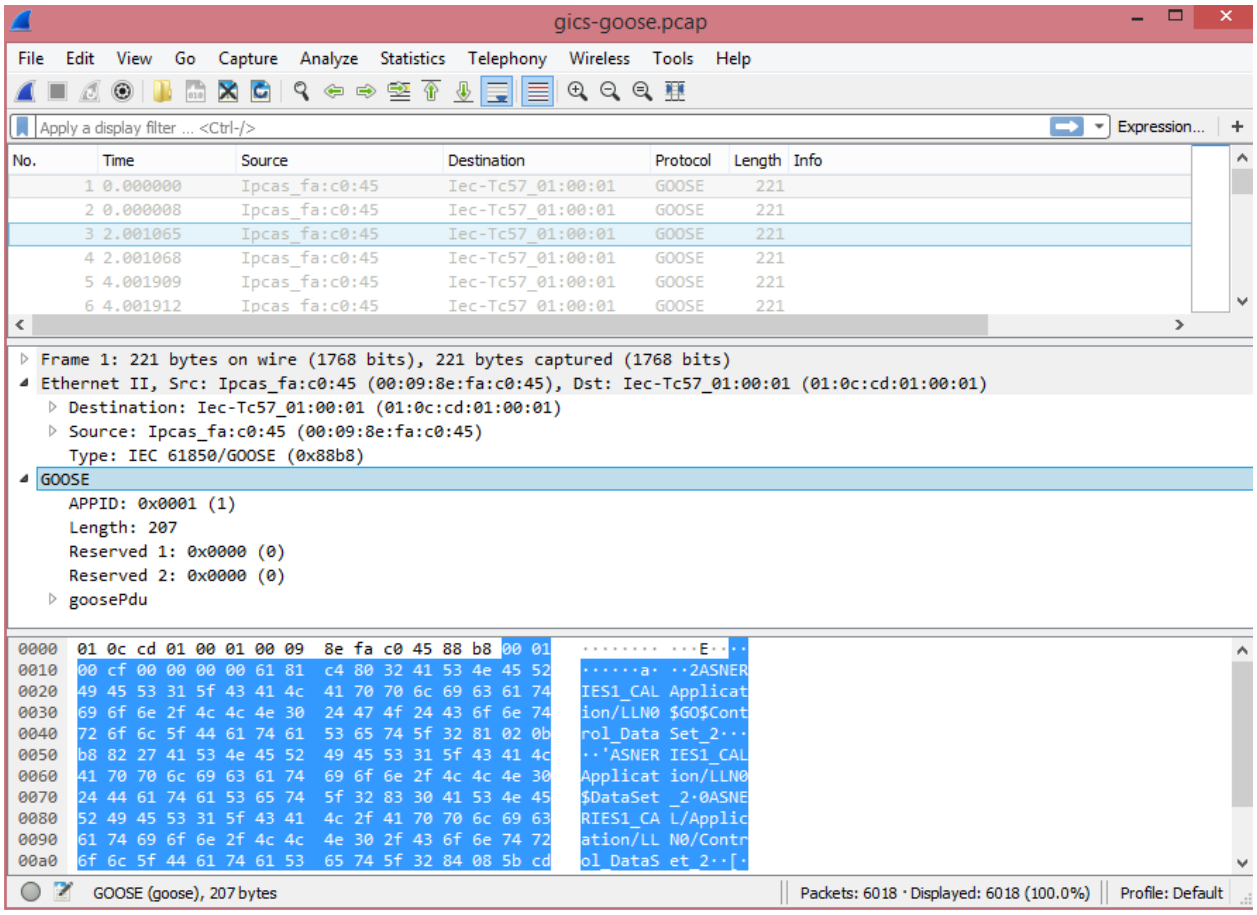

Figure 5: GOOSE in Ethernet frame

Figure 6 below shows the goosePDU and all the fields we have looked for example like APPID, Length, GocbRef, timeAllowedtoLive datSet goID, T, stNum and sqNum. Tshark will be used to process the datagram and extract some interesting fields that will enable identification of traffic in the communication. The descriptions of some of the fields are

Length indicates the total number of bytes in the frame less eight bytes.

goID GOOSE message identification attribute with standard values of GOOSEID 65 octets

datSet Object reference of control block whose values of members shall be transmitted. <LDName>/<LNName>. <CBName> 129 octets

stNum Status number is a counter that increments each time a GOOSE message has been sent with any change in the values of the Data Set.

sqNum Sequence number of the current report and increments each time a GOOSE message is sent.

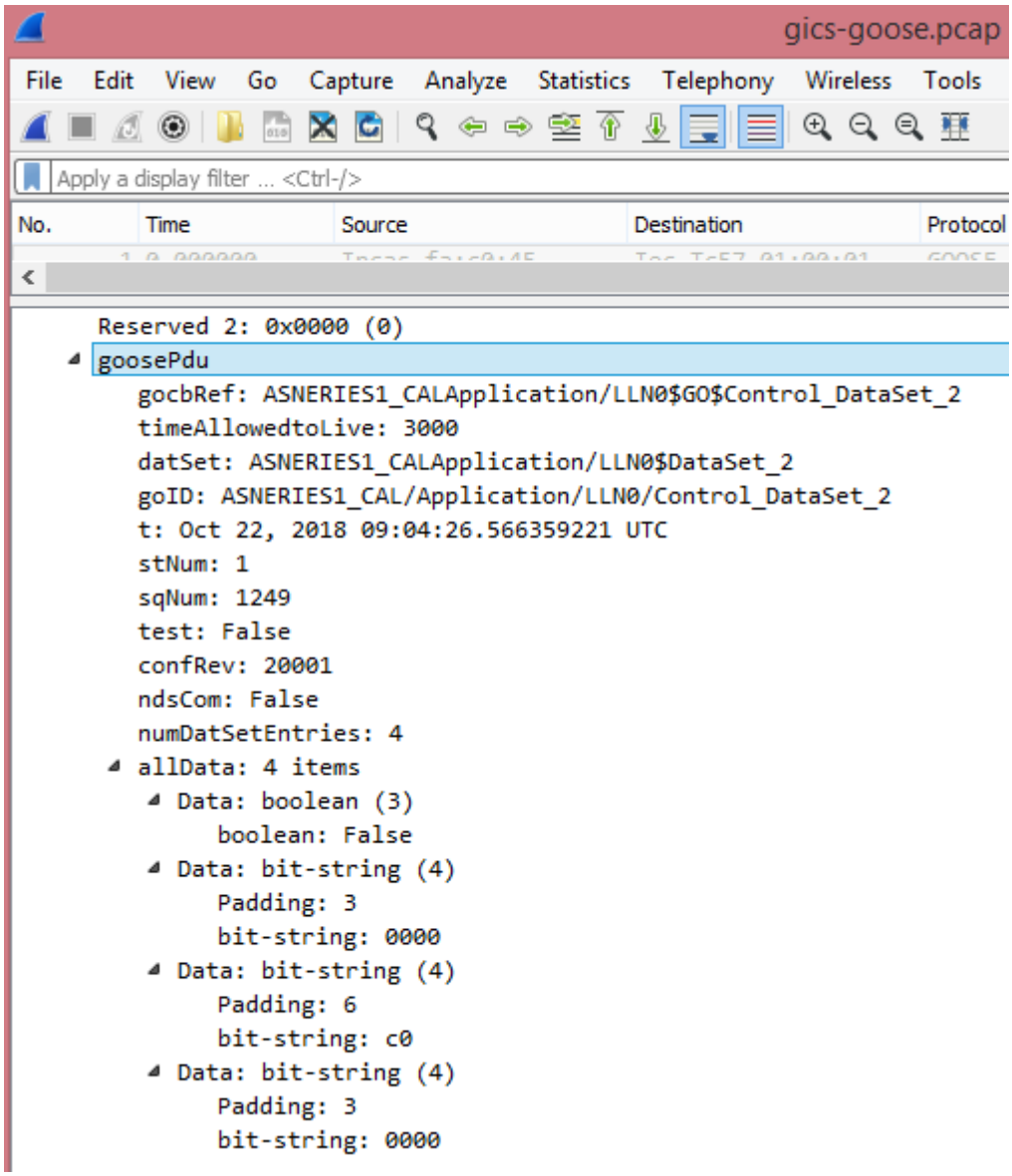

Figure 6: goosePDU

For goose analysis, I will use the fields shown in the example tshark command below.

```
tshark -r gics-goose.pcap -T fields -E separator=";" -e frame.time_relative -e goose.appid -e eth.src -e eth.dst -e goose.gocbRef -e goose.goID –e goose.datSet -e goose.stNum -e goose.sqNum -e goose.t  > gooseEX111.csv
```

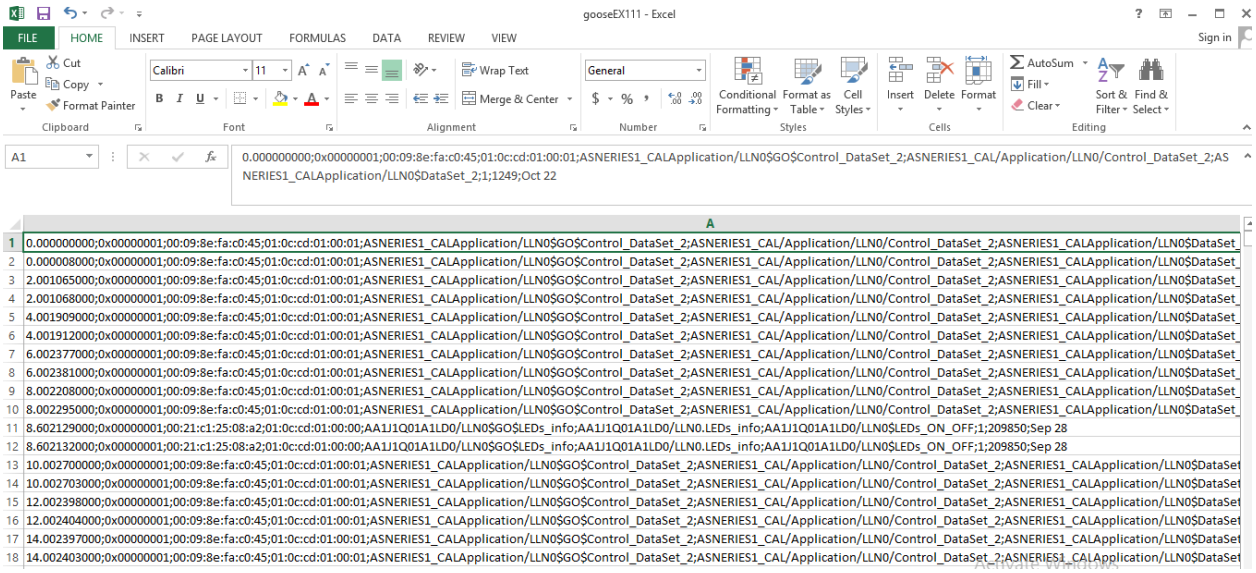

Figure 7: CSV file created using tshark command

As can be seen in Figure 7 above, tshark extracts useful communication that can be processed and analyzed and the data is saved in a CSV file (gooseEX111.csv). Data can be extracted in the command shell and also be processed in shell, but we have saved the information into a CSV file since if you have large data, the data processing process can be cumbersome. Also, a CSV file is able to handle large dataset than the command set for processing and also easier to visualize data. Some of the goose processing performed are separated conversations based on Src Mac, Dst Mac, goCBRef, goID and datsat. Please see Figure 8 below processed fields.

It can be viewed that the first rows 10 rows are identical which represent one conversation as shown in the below extract. Notice the data in goose.gocbRef, goose.goID and goose.dataSet.

frame.time_relative	goose.appid	eth.src	eth.dst	goose.gocbRef	goose.goID	goose.datSet	goose.stNum	goose.sqNum	goose.t
0	0x00000001	00:09:8e:fa:c0:45	01:0c:cd:01:00:01	ASNERIES1_CALApplication/LLN0GOControl_DataSet_2	ASNERIES1_CAL/Application/LLN0/Control_DataSet_2	ASNERIES1_CALApplication/LLN0$DataSet_2	1	1249	Oct 22, 2018 09:04:26.566359221 UTC
0.000008	0x00000001	00:09:8e:fa:c0:45	01:0c:cd:01:00:01	ASNERIES1_CALApplication/LLN0GOControl_DataSet_2	ASNERIES1_CAL/Application/LLN0/Control_DataSet_2	ASNERIES1_CALApplication/LLN0$DataSet_2	1	1249	Oct 22, 2018 09:04:26.566359221 UTC

The extract below represents the second conversation.

frame.time_relative	goose.appid	eth.src	eth.dst	goose.gocbRef	goose.goID	goose.datSet	goose.stNum	goose.sqNum	goose.t
8.602129	0x00000001	00:21:c1:25:08:a2	01:0c:cd:01:00:00	AA1J1Q01A1LD0/LLN0GOLEDs_info	AA1J1Q01A1LD0/LLN0.LEDs_info	AA1J1Q01A1LD0/LLN0$LEDs_ON_OFF	1	209850	Sep 28, 2018 08:39:58.0684651673
8.602132	0x00000001	00:21:c1:25:08:a2	01:0c:cd:01:00:00	AA1J1Q01A1LD0/LLN0GOLEDs_info	AA1J1Q01A1LD0/LLN0.LEDs_info	AA1J1Q01A1LD0/LLN0$LEDs_ON_OFF	1	209850	Sep 28, 2018 08:39:58.0684651673

F16 ASNERIES1_CAL/Application/LLN0/Control_DataSet_2

	A	B	C	D	E	F
1	frame.time_relative	goose.appid	eth.src	eth.dst	goose.gocbRef	goose.goID
2	0	0x00000001	00:09:8e:f	01:0c:cd:0	ASNERIES1_CALApplication/LLN0GOControl_DataSet_2	ASNERIES1_CAL/Application/LLN0/Control_DataSet_2
3	0.000008	0x00000001	00:09:8e:f	01:0c:cd:0	ASNERIES1_CALApplication/LLN0GOControl_DataSet_2	ASNERIES1_CAL/Application/LLN0/Control_DataSet_2
4	2.001065	0x00000001	00:09:8e:f	01:0c:cd:0	ASNERIES1_CALApplication/LLN0GOControl_DataSet_2	ASNERIES1_CAL/Application/LLN0/Control_DataSet_2
5	2.001068	0x00000001	00:09:8e:f	01:0c:cd:0	ASNERIES1_CALApplication/LLN0GOControl_DataSet_2	ASNERIES1_CAL/Application/LLN0/Control_DataSet_2
6	4.001909	0x00000001	00:09:8e:f	01:0c:cd:0	ASNERIES1_CALApplication/LLN0GOControl_DataSet_2	ASNERIES1_CAL/Application/LLN0/Control_DataSet_2
7	4.001912	0x00000001	00:09:8e:f	01:0c:cd:0	ASNERIES1_CALApplication/LLN0GOControl_DataSet_2	ASNERIES1_CAL/Application/LLN0/Control_DataSet_2
8	6.002377	0x00000001	00:09:8e:f	01:0c:cd:0	ASNERIES1_CALApplication/LLN0GOControl_DataSet_2	ASNERIES1_CAL/Application/LLN0/Control_DataSet_2
9	6.002381	0x00000001	00:09:8e:f	01:0c:cd:0	ASNERIES1_CALApplication/LLN0GOControl_DataSet_2	ASNERIES1_CAL/Application/LLN0/Control_DataSet_2
10	8.002208	0x00000001	00:09:8e:f	01:0c:cd:0	ASNERIES1_CALApplication/LLN0GOControl_DataSet_2	ASNERIES1_CAL/Application/LLN0/Control_DataSet_2
11	8.002295	0x00000001	00:09:8e:f	01:0c:cd:0	ASNERIES1_CALApplication/LLN0GOControl_DataSet_2	ASNERIES1_CAL/Application/LLN0/Control_DataSet_2
12	8.602129	0x00000001	00:21:c1:2	01:0c:cd:0	AA1J1Q01A1LD0/LLN0GOLEDs_info	AA1J1Q01A1LD0/LLN0.LEDs_info
13	8.602132	0x00000001	00:21:c1:2	01:0c:cd:0	AA1J1Q01A1LD0/LLN0GOLEDs_info	AA1J1Q01A1LD0/LLN0.LEDs_info
14	10.0027	0x00000001	00:09:8e:f	01:0c:cd:0	ASNERIES1_CALApplication/LLN0GOControl_DataSet_2	ASNERIES1_CAL/Application/LLN0/Control_DataSet_2
15	10.002703	0x00000001	00:09:8e:f	01:0c:cd:0	ASNERIES1_CALApplication/LLN0GOControl_DataSet_2	ASNERIES1_CAL/Application/LLN0/Control_DataSet_2
16	12.002398	0x00000001	00:09:8e:f	01:0c:cd:0	ASNERIES1_CALApplication/LLN0GOControl_DataSet_2	ASNERIES1_CAL/Application/LLN0/Control_DataSet_2

	F	G	H	I	J	K
	goose.goID	goose.datSet	goose.stNum	goose.sqNum	goose.t	
	ASNERIES1_CAL/Application/LLN0/Control_DataSet_2	ASNERIES1_CALApplication/LLN0$DataSet_2	1	1249	Oct 22, 2018 09:04:26.5	
	ASNERIES1_CAL/Application/LLN0/Control_DataSet_2	ASNERIES1_CALApplication/LLN0$DataSet_2	1	1249	Oct 22, 2018 09:04:26.5	
	ASNERIES1_CAL/Application/LLN0/Control_DataSet_2	ASNERIES1_CALApplication/LLN0$DataSet_2	1	1250	Oct 22, 2018 09:04:26.5	
	ASNERIES1_CAL/Application/LLN0/Control_DataSet_2	ASNERIES1_CALApplication/LLN0$DataSet_2	1	1250	Oct 22, 2018 09:04:26.5	
	ASNERIES1_CAL/Application/LLN0/Control_DataSet_2	ASNERIES1_CALApplication/LLN0$DataSet_2	1	1251	Oct 22, 2018 09:04:26.5	
	ASNERIES1_CAL/Application/LLN0/Control_DataSet_2	ASNERIES1_CALApplication/LLN0$DataSet_2	1	1251	Oct 22, 2018 09:04:26.5	
	ASNERIES1_CAL/Application/LLN0/Control_DataSet_2	ASNERIES1_CALApplication/LLN0$DataSet_2	1	1252	Oct 22, 2018 09:04:26.5	
	ASNERIES1_CAL/Application/LLN0/Control_DataSet_2	ASNERIES1_CALApplication/LLN0$DataSet_2	1	1252	Oct 22, 2018 09:04:26.5	
	ASNERIES1_CAL/Application/LLN0/Control_DataSet_2	ASNERIES1_CALApplication/LLN0$DataSet_2	1	1253	Oct 22, 2018 09:04:26.5	
	ASNERIES1_CAL/Application/LLN0/Control_DataSet_2	ASNERIES1_CALApplication/LLN0$DataSet_2	1	1253	Oct 22, 2018 09:04:26.5	
	AA1J1Q01A1LD0/LLN0.LEDs_info	AA1J1Q01A1LD0/LLN0$LEDs_ON_OFF	1	209850	Sep 28, 2018 08:39:58.(	
	AA1J1Q01A1LD0/LLN0.LEDs_info	AA1J1Q01A1LD0/LLN0$LEDs_ON_OFF	1	209850	Sep 28, 2018 08:39:58.(	
	ASNERIES1_CAL/Application/LLN0/Control_DataSet_2	ASNERIES1_CALApplication/LLN0$DataSet_2	1	1254	Oct 22, 2018 09:04:26.5	
	ASNERIES1_CAL/Application/LLN0/Control_DataSet_2	ASNERIES1_CALApplication/LLN0$DataSet_2	1	1254	Oct 22, 2018 09:04:26.5	
	ASNERIES1_CAL/Application/LLN0/Control_DataSet_2	ASNERIES1_CALApplication/LLN0$DataSet_2	1	1255	Oct 22, 2018 09:04:26.5	

Figure 8: GOOSE Processing using Excel

Learning and Detection (Goose Communication)

Github Link: https://github.com/simon-wachiuri/probability-distribution-GOOSE

A set of GOOSE dataset with unique fields obtained from the previous step is used as the training dataset with a defined profile based on probability density function (PDF). The continuous dataset is discretised with bins. The following terminologies are critical in the implementation of this learning phase:

1. **Bins**

Binning or discretisation is the process of transforming numerical variables into categorical counterparts to improve accuracy of the predictive models by reducing the noise or non-linearity. Bins allow easy identification of outliers, invalid and missing values of numerical variables. Unsupervised binning methods largely fall into Equal Width and Equal Frequency. Equal width is the simplest binning approach to partition the range of the variable into k equal-width intervals and in equal-frequency binning, it is obtained by dividing the range of the variable into intervals that contain (approximately) equal number of points.

In the context of our dataset, dataset207 (named according to the **GOOSE LENGTH** of the csv dataset obtained from the Pcap file) is observed to have the column delta_T range between 0.0 to 8.0 seconds. It is important to note that the column **delta_T** is obtained from the calculation of the difference in the rows of the **EPOCH_TIME** column. A snippet of this data is shown in Figure 9.

EPOCH_TIME	delta_T	GOOSE_ID	GOOSE DAT.SET	GOOSE LENGTH
1540201549	2.001060009	ASNERIES1_CA	ASNERIES1_CALApplicat	207
1540201551	0	ASNERIES1_CA	ASNERIES1_CALApplicat	207
1540201551	2.000840187	ASNERIES1_CA	ASNERIES1_CALApplicat	207
1540201553	1.00E-05	ASNERIES1_CA	ASNERIES1_CALApplicat	207
1540201553	2.000459909	ASNERIES1_CA	ASNERIES1_CALApplicat	207
1540201555	0	ASNERIES1_CA	ASNERIES1_CALApplicat	207
1540201555	1.999830008	ASNERIES1_CA	ASNERIES1_CALApplicat	207
1540201557	8.99E-05	ASNERIES1_CA	ASNERIES1_CALApplicat	207
1540201557	2.000400066	ASNERIES1_CA	ASNERIES1_CALApplicat	207
1540201559	1.00E-05	ASNERIES1_CA	ASNERIES1_CALApplicat	207
1540201559	1.999690056	ASNERIES1_CA	ASNERIES1_CALApplicat	207
1540201561	1.00E-05	ASNERIES1_CA	ASNERIES1_CALApplicat	207
1540201561	1.999989986	ASNERIES1_CA	ASNERIES1_CALApplicat	207
1540201563	1.00E-05	ASNERIES1_CA	ASNERIES1_CALApplicat	207
1540201563	4.000169992	ASNERIES1_CA	ASNERIES1_CALApplicat	207

Figure 9: A snippet of the dataset 207 with all the columns

In constructing our equal bins, we take a range of -1 to 10 seconds so that we are able to capture the range of values available. Analysing the data further demonstrates that the delta_T is largely concentrated between 0-4 seconds. Based on the IEC 61850 standard, this matches the normal behaviour of GOOSE communication. The algorithm on dataset_207 based on this property generates ten bins of equal width i.e. the ten bins range from 0.0-10.0 In this case, GOOSE heartbeat messages can be said to be send every 4s in the transmission. A value lower than 4s shows that events are occurring in the transmission. The microseconds

represent shortest retransmission time after the events. Values higher than 4s represent retransmission with no events for a long time. The bins generated by the algorithm are shown in the snippet of Figure 10 together with their percentage of occurrence in the dataset.

```
Bins                        Percentages:
 0.000001-2.000000     33.187773
-1-0.000000           31.669786
2.000001-3.000000     23.206488
4.000001-5.000000     10.147640
3.000001-4.000000      1.559576
6.000001-7.000000      0.187149
8.000001-9.000000      0.020794
5.000001-6.000000      0.020794
9.000001-10.000000     0.000000
7.000001-8.000000      0.000000
Name: delta_T, dtype: float64
```

Figure 10: A snippet of the bins with their percentage of distribution

2. Probability distribution

A probability distribution is a table or an equation that links each outcome of a statistical experiment with its probability of occurrence. Knowing the probability distribution for a variable can help to calculate moments of the distribution, like the mean and variance, but can also be useful for other more general considerations, like determining whether an observation is unlikely or very unlikely and might be an outlier or anomaly.

In the case of data207, our range of values are distributed against the bins. The description of the distribution is based on the occurrence of the values of **delta_T** in the bins. Measures of central tendencies have also been computed on the dataset to obtain the mean and the standard deviation for standardising the data and determine a distribution that can be provided on a histogram.

The distribution of the original data is shown in Figure 11 while the distribution based on the standardised data is shown in Figure 12. The mean and the standard deviation of data (obtained from the array of the dataframe).

The histograms gives us a visual insight of the coarseness of the distribution and, in turn, how well the density of the observations is plotted.

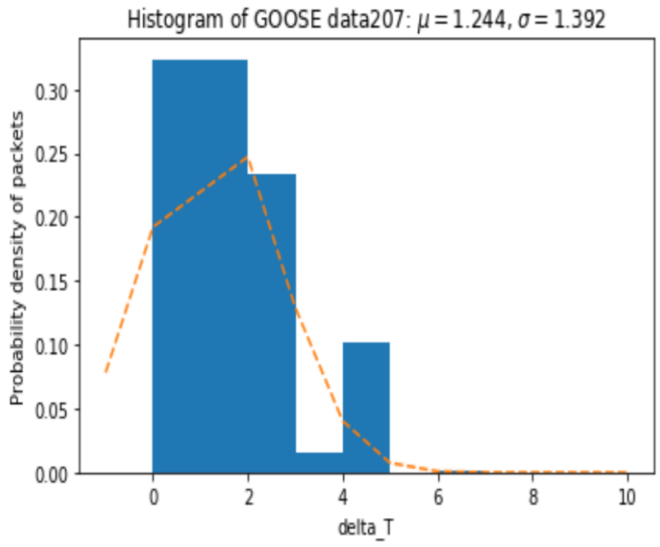

Figure 11: The distribution of the original data

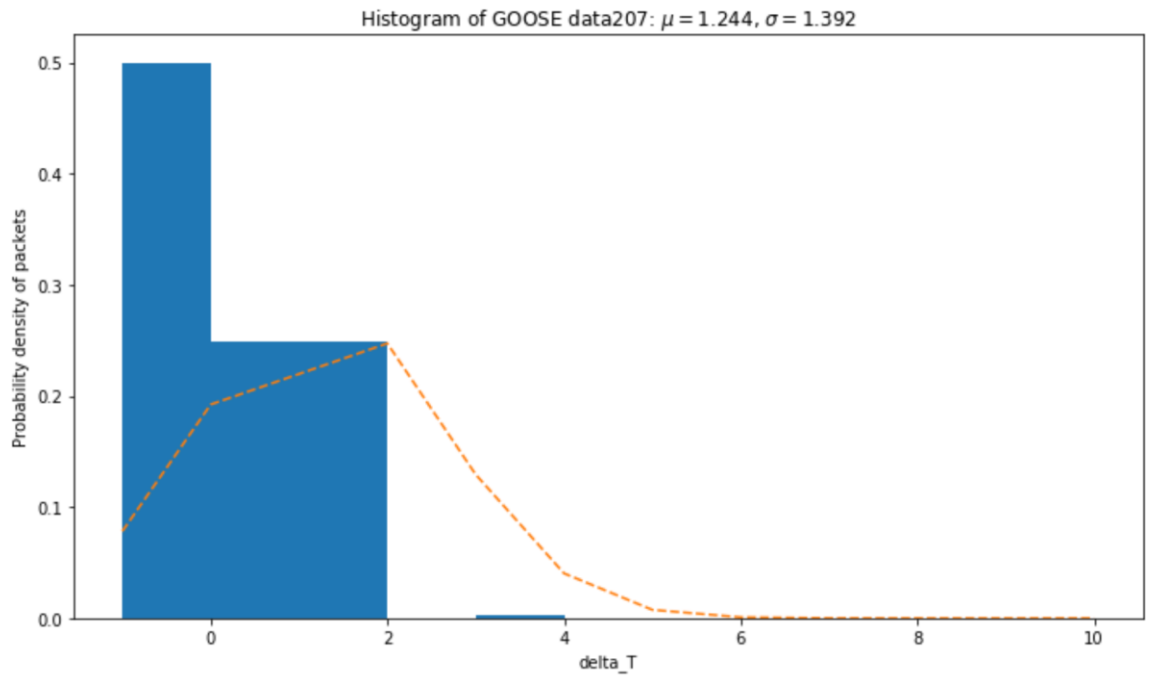

Figure 12: the distribution based on the standardised data

3. Euclidean Distance and Detection

Euclidean distance is the distance between two points in Euclidean space. Euclidean distance is calculated as the square root of the sum of the squared differences between a point *a* and point *b* across all input attributes *i*.

$$Euclidean\ distance\ (a,b) = \sqrt{\sum_{i=1}^{n} (a_i - b_i)}$$

In the case of this study, the Euclidean distance is used as a hinge point within which a deviation on the dataset is set. Any new datasets with varying sizes and delta_T can be assessed if they are a normal dataset for GOOSE communication or not based on the calculated Euclidean distance. Due to the nature of our dataset207, size is assumed as a "similar-width" parameter to the delta_T for demonstration. An algorithm that takes on the dataframe to calculate the Euclidean distance is applied to the delta_T and the size. This produces a Euclidean distance of **14268.901156384632.** This Euclidean distance value is then taken as a reference Euclidean distance which any dataset fed into the overall algorithm checks to determine how much it deviates from the determined distance and by how much in percentage it deviates. This is what executes the **detection** comparison to determine an "attack" or the deviation of the new dataset to the reference dataset207.

As an example of implementation, execution of the reference dataset produces its distribution as described in the previous section and outputs the Euclidean distance as shown below:

```
Euclidean Distance: 14268.901156384632
The data matches 100%, no deviation
```

Since this is selected as the reference Euclidean distance, on checking the deviation, a message that shows “The data matches 100%, no deviation” to indicate that this is a full match of the data based on the reference distance (named as *stand_euclidean* in the Python code).

Assuming we apply a new dataset (e.g. another dataset with the same size) to the algorithm with the determined Euclidean distance for reference, we obtain some variation. For example, let us use a dataset (**dataset207- spoofing.csv**) that has the same size as dataset207 but not the same columns, we definitely obtain a different distribution as well as the Euclidean distance and we also get an output of how much that dataset deviates from dataset207. The output of this dataset appears as below:

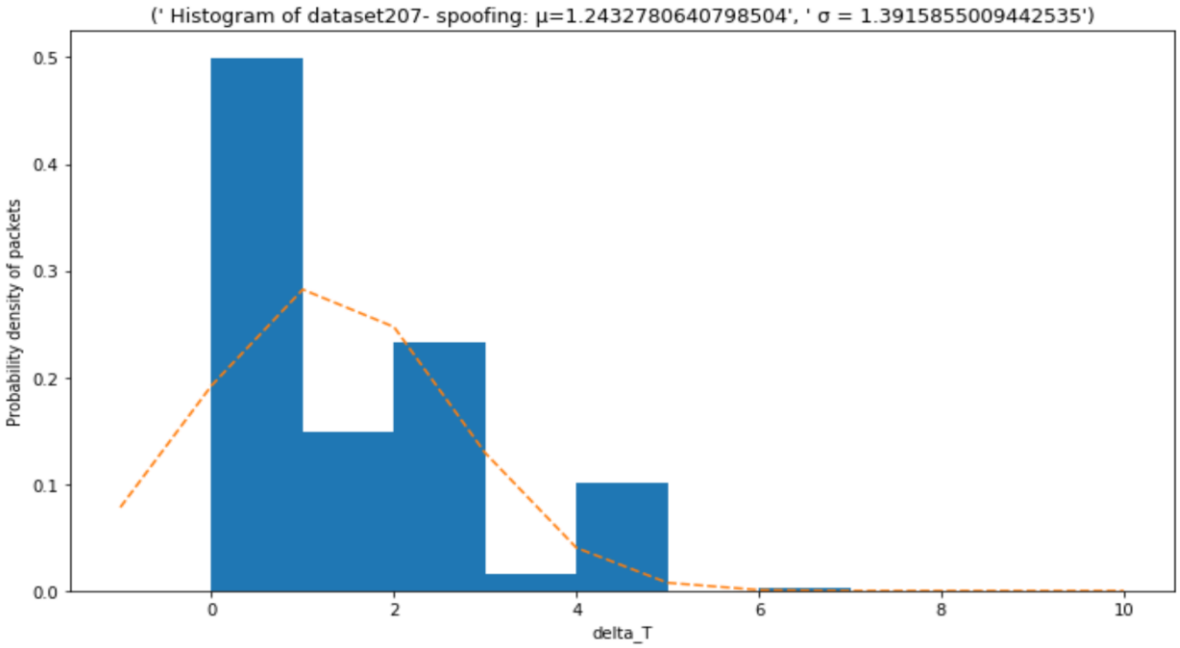

Figure 13: The distribution on dataset207- spoofing

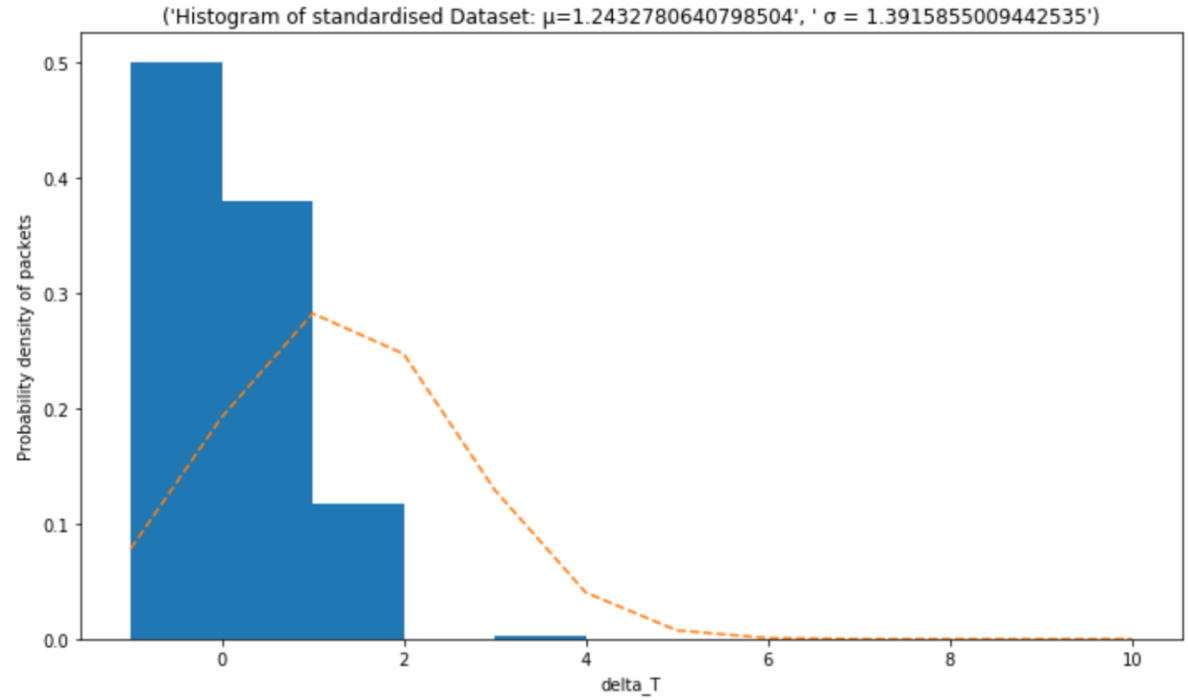

Figure 13:The distribution standardised on dataset207- spoofing

The deviation from the reference Euclidean is as described shown here:

Euclidean Distance: 14287.02533022423
The deviation is by a positive margin of: 0.1270187076142769 %

Assuming we now apply a dataset that has a **dataset of 20 percent** of the columns of the original dataset, we obtain the following outputs:

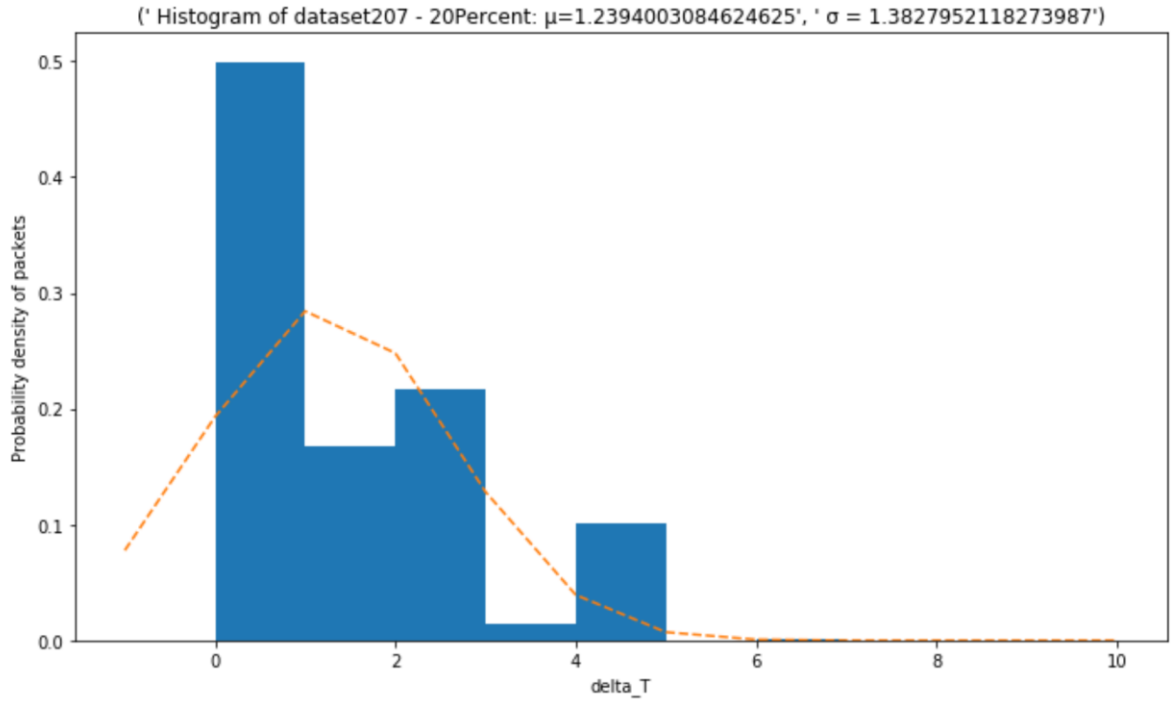

Figure 14: The distribution on dataset - 20 percent

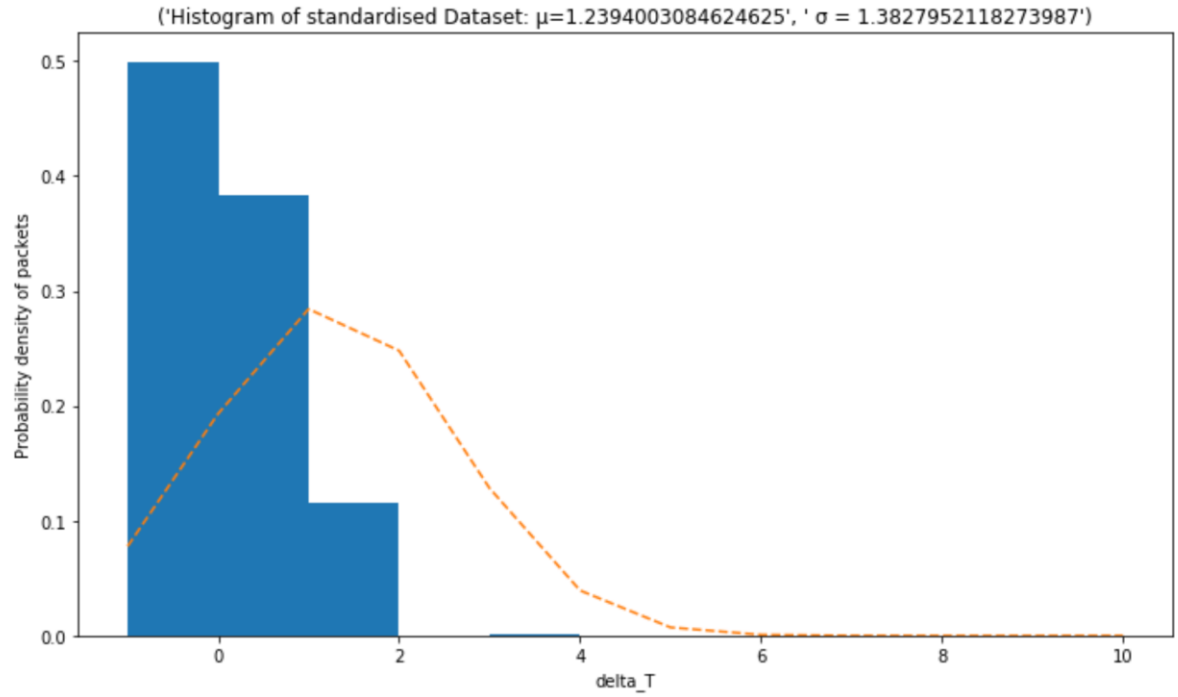

Figure 15: The distribution standardised on dataset - 20 percent

The deviation from the reference Euclidean is as described shown here:

```
Euclidean Distance: 6503.61416225709
The deviation is by a positive margin of: 54.42105813910525 %
```

The original data had two conversations of GOOSE communication and looking at **dataset148**, (named according to the **GOOSE LENGTH** of the csv dataset obtained from the Pcap file) it is observed to have the column delta_T range between 0.0 to 9.9 seconds. A snippet of the data is shown in Figure 16 below together with the distribution within the bins i.e. Figure 17.

EPOCH_TIME	delta_T	GOOSE_ID	GOOSE DAT.SET	GOOSE LENGTH
1540201558	0	AA1J1Q01A1LD0/LI	AA1J1Q01A1LD0/LL	148
1540201558	9.900418997	AA1J1Q01A1LD0/LI	AA1J1Q01A1LD0/LL	148
1540201568	0	AA1J1Q01A1LD0/LI	AA1J1Q01A1LD0/LL	148
1540201568	9.900635004	AA1J1Q01A1LD0/LI	AA1J1Q01A1LD0/LL	148
1540201578	0	AA1J1Q01A1LD0/LI	AA1J1Q01A1LD0/LL	148
1540201578	9.900574923	AA1J1Q01A1LD0/LI	AA1J1Q01A1LD0/LL	148
1540201588	0	AA1J1Q01A1LD0/LI	AA1J1Q01A1LD0/LL	148
1540201588	9.900579929	AA1J1Q01A1LD0/LI	AA1J1Q01A1LD0/LL	148
1540201598	7.15E-06	AA1J1Q01A1LD0/LI	AA1J1Q01A1LD0/LL	148
1540201598	9.900631905	AA1J1Q01A1LD0/LI	AA1J1Q01A1LD0/LL	148
1540201608	7.15E-06	AA1J1Q01A1LD0/LI	AA1J1Q01A1LD0/LL	148
1540201608	9.900613785	AA1J1Q01A1LD0/LI	AA1J1Q01A1LD0/LL	148
1540201617	0	AA1J1Q01A1LD0/LI	AA1J1Q01A1LD0/LL	148
1540201617	9.900658131	AA1J1Q01A1LD0/LI	AA1J1Q01A1LD0/LL	148

Figure 16: A snippet of the dataset 148 with all the columns

Bins	Percentages:
9.000001-10.000000	49.917219
-1-0.000000	46.192053
0.000001-1.000000	3.890728
8.000001-9.000000	0.000000
7.000001-8.000000	0.000000
6.000001-7.000000	0.000000
5.000001-6.000000	0.000000
4.000001-5.000000	0.000000
3.000001-4.000000	0.000000
2.000001-3.000000	0.000000
1.000001-2.000000	0.000000

Figure 17: A snippet of the bins with their percentage of distribution

Assuming we now apply **dataset 148** to the columns of the original dataset in this case consider dataset 207, we obtain the following outputs:

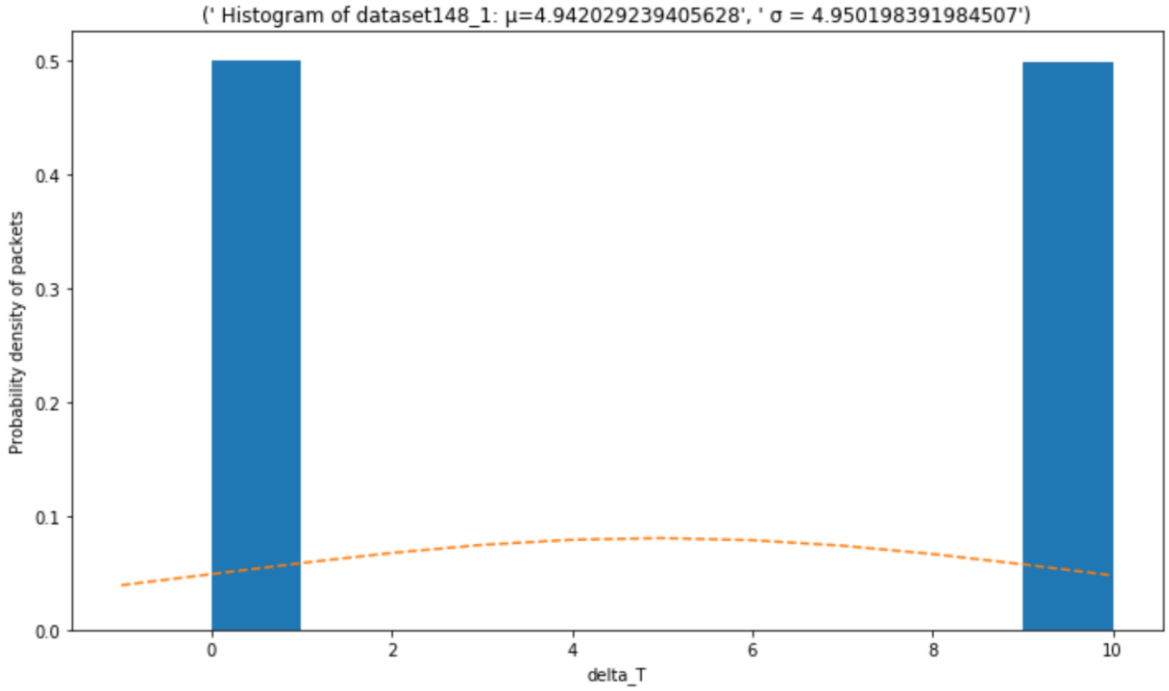

Figure 18: The distribution on dataset148_1

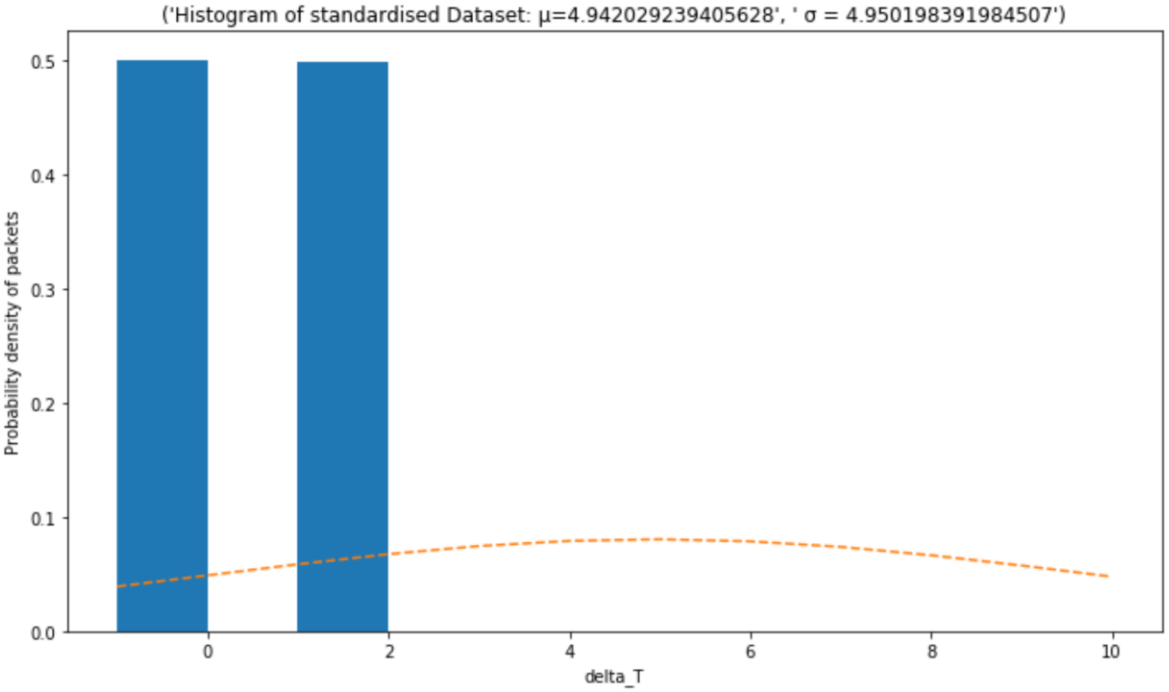

Figure 19: The distribution standardised on dataset148_1

The deviation from the reference Euclidean is as described shown here:

Euclidean Distance: 4975.140767295833
The deviation is by a positive margin of: 65.13297896755208 %

Recommendations

>> Quality of the datasets/different dataset that is not biased towards a bigger percentage of delta_T

>> Determination of statistical analysis using an approach different from the Euclidean distance

>> Different bin selection converging to non-zero delta_T

Future Work

>> Advanced studies on the implementation
>> Deeper research on the implementation of the Euclidean distance on the dataset eliminating the assumption of the computation used at this time.

Bibliography

Crotti, M., Gringoli, F., & Salgarelli, L. (2007). Traffic Classification Through Simple Statistical Fingerprinting. *Computer Communication Review* (p. 10). Brescia, Italy: AGC Computer Communication Review.

Karagiannis , T., Papagiannaki, K., & Faloutsos, M. (2005). BLINC: Multilevel Traffic Classification in the Dark. *ACM SIGCOMM Computer Communication Review* (p. 12). Philadelphia, Pennsylvania: SIGCOMM.

Kwon , Y., Kim, H. K., Lim, Y. H., & Lim, J. I. (2015). A behavior-based intrusion detection technique for smart grid infrastructure. *IEEE Eindhoven PowerTec*, 6.

Matoušek, P., Ondřej , R., & Matěj , G. (2019). Security Monitoring of IoT Communication Using Flows. *6th Conference on the Engineering of Computer Based Systems* (p. 9). Bucharest: Association for Computing Machinery.

Slavica , V., Rakas, B., & Stojanović, M. D. (2020). A Review of Research Work on Network-Based SCADA Intrusion Detection Systems. *IEEE Access*, 26.

Observations

One had to use -1 0 bin range as the zeros were not being captured when profiling the data.

4. Types of Attacks - Findings

Spoofing attack

Spoofing attack utilizes a GOOSE exploit where an attacker publishes false layer 2 packets and devices on the receiving side mistakenly believe they are receiving valid (true) packets sent by a trusted or secured entity. This attack is possible due to the unencrypted & unauthenticated nature of GOOSE messages, owing to the latency issues on IED devices. The attacker publishes false packets and subscriber IEDs mistakenly believe they receiving packets send by a trusted or secured publisher or entity. This attack is possible due to the unencrypted & unauthenticated nature of GOOSE messages, owing to the latency issues on IED devices e.g. IEC 61850-5 specifies a 4ms maximum delay for GOOSE messages related to breaker trip functions.

A practical spoof attack can include; first, monitoring packets on the physical ports looking for GOOSE messages based on Ether-type identification e.g. Ethernet frames with specific GOOSE Ether-type of 0x88B8. Secondly, decode GOOSE using Abstract Syntax Notation One (ASN1) and looking for three specific GOOSE fields namely stNum, sqNum, and the Boolean values inside the data sets. While keeping the sequence for the different counters and timers, change any Boolean value inside the dataset, if the value is true the code overwrites a false and vice versa. Lastly, the packet is decoded using Basic Encoding Rules (BER) and send the spoofed messages through a physical port obfuscating the source MAC address.

Results/Attack Findings:

The results indicated in figure 9 show that the true values belong to the timestamps of 193 and 198. One notice that the spoofed messages are on the events 194 to 197 and the time to generate spoofed GOOSE messages is less than 1ms (time stamp of the packets 193 and 195). In a default GOOSE configuration, where the messages are sent at 1 second intervals during steady state, the attack could inject hundreds of false GOOSE messages before the next valid datagram reaches the IED.

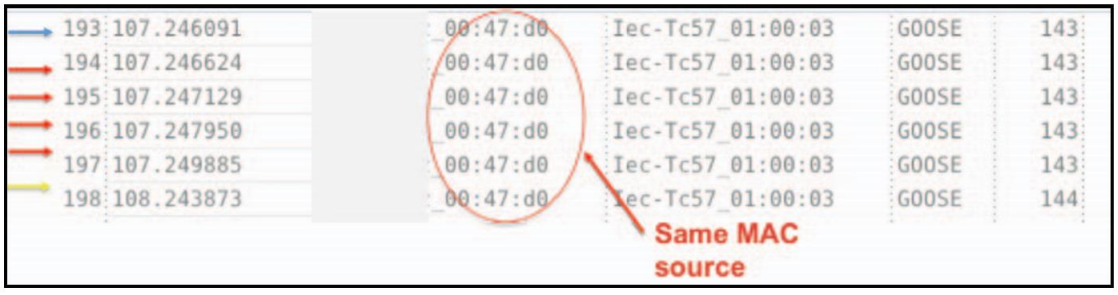

Figure 20: Wireshark Capture of Spoofed GOOSE Communication

The process of modifying can be summarized in the datagram extract below. The first datagram is a valid message. The middle message created by an attacker shows the change of stNum, which resets the SqNum in the cloned packet. The last (rightmost) message is the next valid message which keeps the old number sequence meaning it is out of sequence. By decoding the message, the attacker changes the Boolean data value from False to True thus creating the attack datagram. This simple attack enables malware to control IEC 61850-

enabled IEDS and cause outages that range from a single feeder on up to even a regional smart grid.

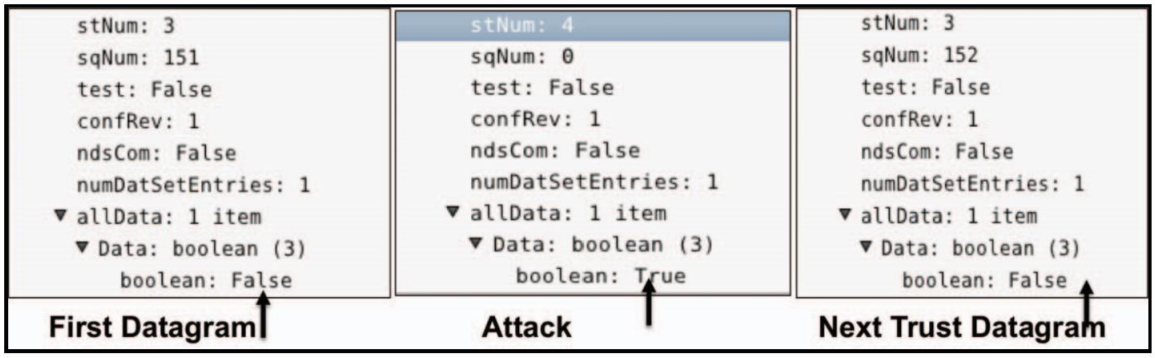

Figure 21: GOSE communication exploit

My comments to remember process on CSV remove for main report (insert 220 to stNUM filed from row 568-599 and change the delta times in the rows from 0 to 0.0001 and row 589 from 9.900229931 to 0.009900229931)

The extracted CSV file called dataset 148- spoofing attack, it can be observed from row 568-599 the goose_stNUM field is 220. This is a higher number packet than the previous packet meaning the attacker was successful in taking over the traffic and he can spoof messages he needs. This can be reaffirmed by the checking on the delta Times which are just below 2 minutes. (Q to Prof do I need this last paragraph to tie in the CSV with a picture of a few rows of the CSV or just leave it.)