# Configuration Manual

Industrial Internship
MSc Cybersecurity

Adil Mustafa Khokhawala
Student ID: 22144188

School of Computing
National College of Ireland

Supervisor: Jawad Salahuddin

**National College of Ireland**

**MSc Project Submission Sheet**

**School of Computing**

**Student Name:** Adil Mustafa Khokhawala

**Student ID:** 22144188

**Programme:** MSc Cybersecurity **Year:** 2023

**Module:** Industrial Internship

**Lecturer:** Jawad Salahuddin

**Submission Due Date:** 05-01-2024

**Project Title:** Enhancing LoRaWAN security: Authentication strategies to mitigate rogue device infiltration

**Word Count:** 1299 **Page Count:** 16

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

**Signature:** Adil Mustafa Khokhawala

**Date:** 04-01-2024

**PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST**

| | |
|---|---|
| Attach a completed copy of this sheet to each project (including multiple copies) | □ |
| **Attach a Moodle submission receipt of the online project submission,** to each project (including multiple copies). | □ |
| **You must ensure that you retain a HARD COPY of the project,** both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer. | □ |

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

| **Office Use Only** | |
|---|---|
| Signature: | |
| Date: | |
| Penalty Applied (if applicable): | |

# Configuration Manual

Adil Mustafa Khokhawala
Student ID: 22144188

## 1 Introduction:

This document has been prepared to discuss the steps involved in the implementation of the conducted research. In this project, we have created a Lora network prototype in MATLAB and demonstrated an authentication method for preventing rogue devices from penetrating the network. Additionally, a pilot simulation has been done for the signal parameters of 868 MHz and channel bandwidth of 125 KHz has been shown. To simulate the Poc of the solution different Simulink libraries, Channel blocks and C programming language were used for encryption. This configuration manual gives a detailed synopsis to set up the simulation environment and execute the project successfully.

## 2 Simulation Set up:

**Step 1:** The first step is to download and install the simulation environment. We have used MATLAB version ***R2023a*** for this research. The MATLAB version can be downloaded using this link: https://matlab.mathworks.com/

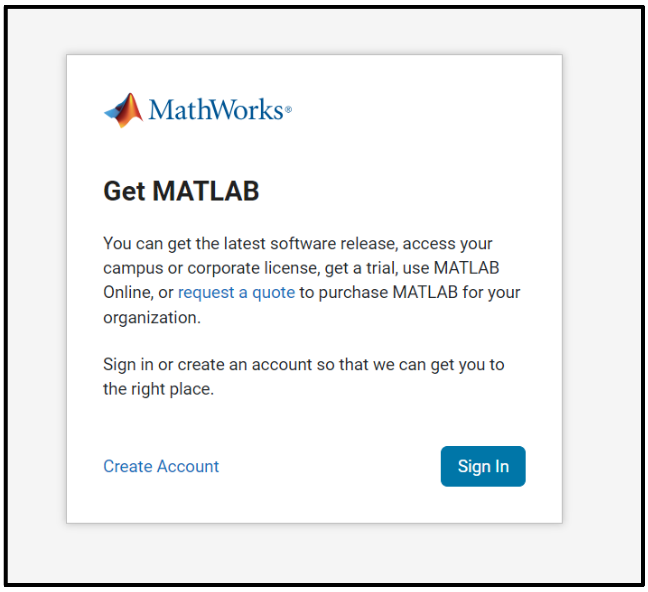


**Figure 1: Sign In**

**Step 2:** After completing the account creation, and signing in process, we need to download the desktop client for MATLAB. The installer can be downloaded by clicking on the button for the R2023a version.

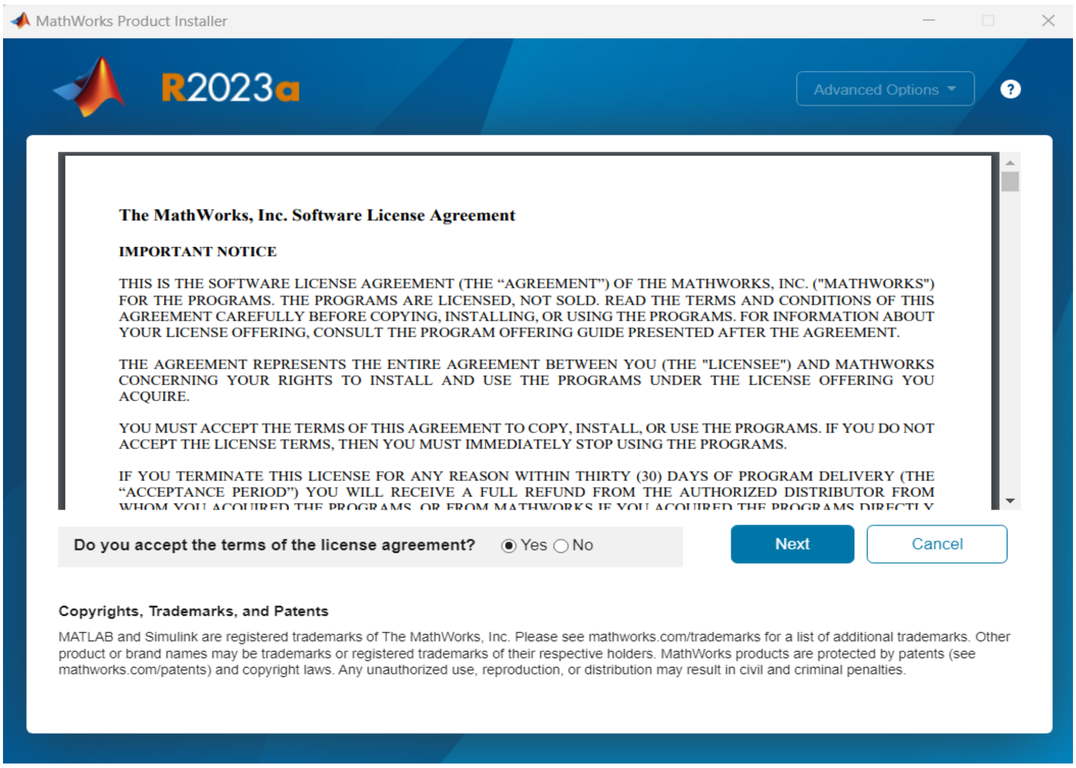


**Figure 2: License Agreement**

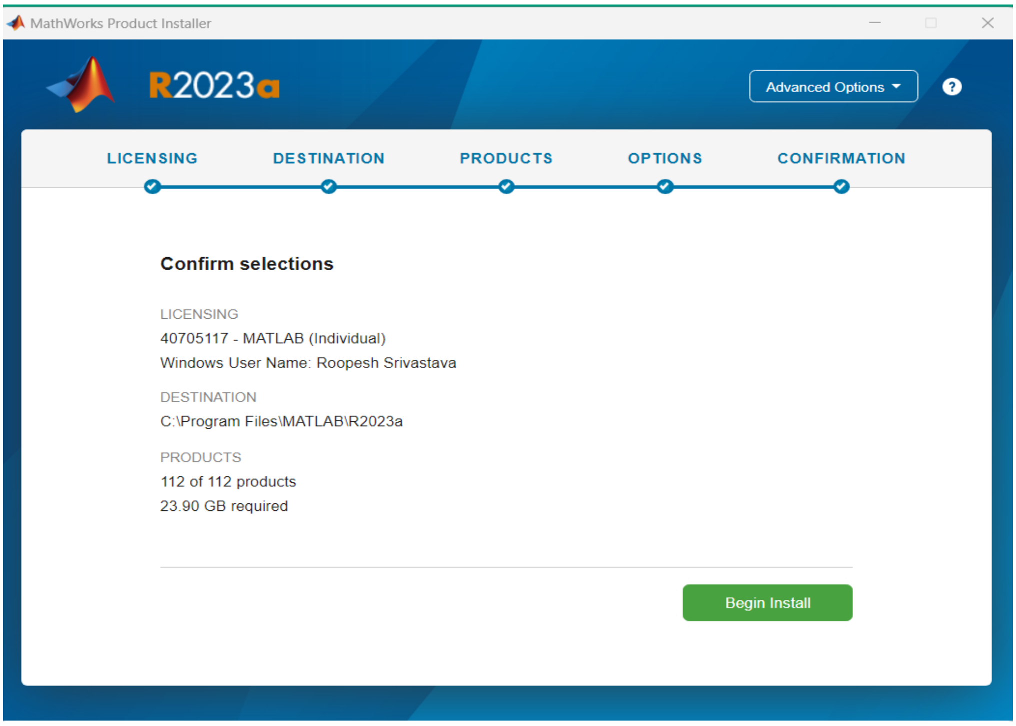


**Figure 3: Begin Installation**

**Step 3:** Upon installing MATLAB, we select all the default options and install all the Simulink libraries in the list. This will begin the installation.

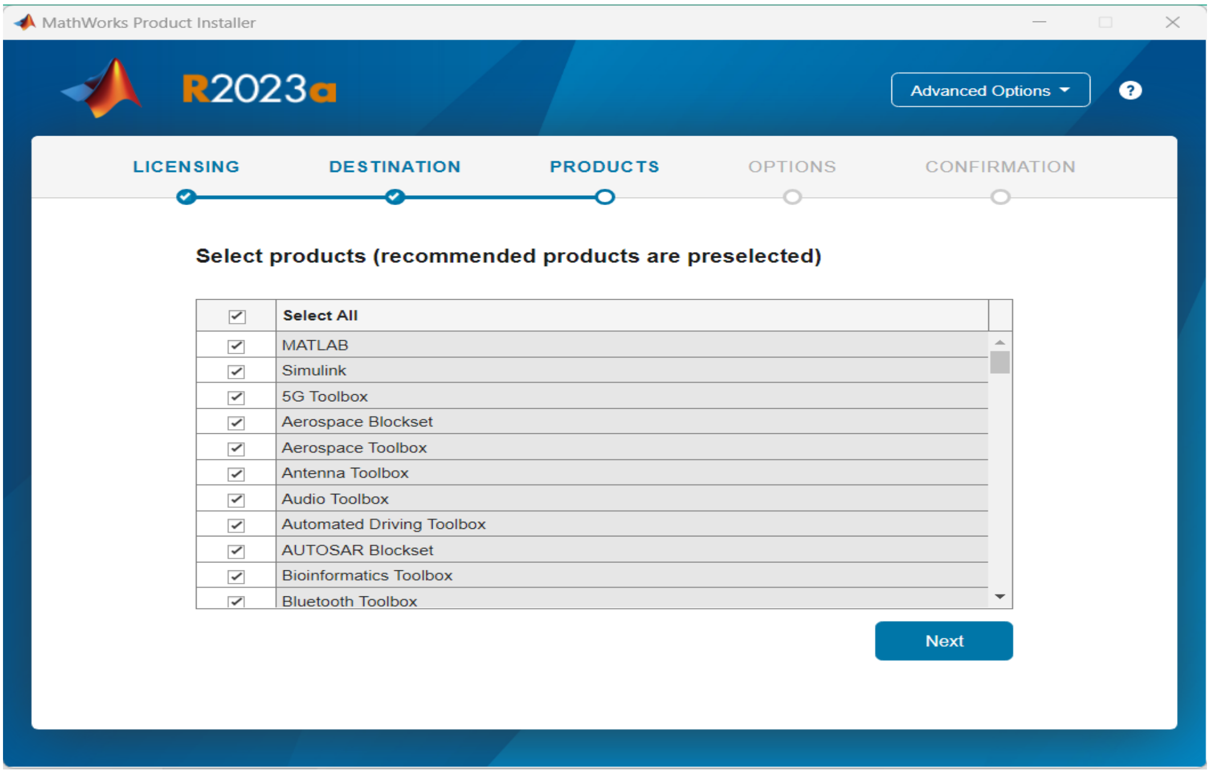


**Figure 4 : Product List**

**Step 4:** Launch the MATLAB R2023a client and boot up the software for simulation.

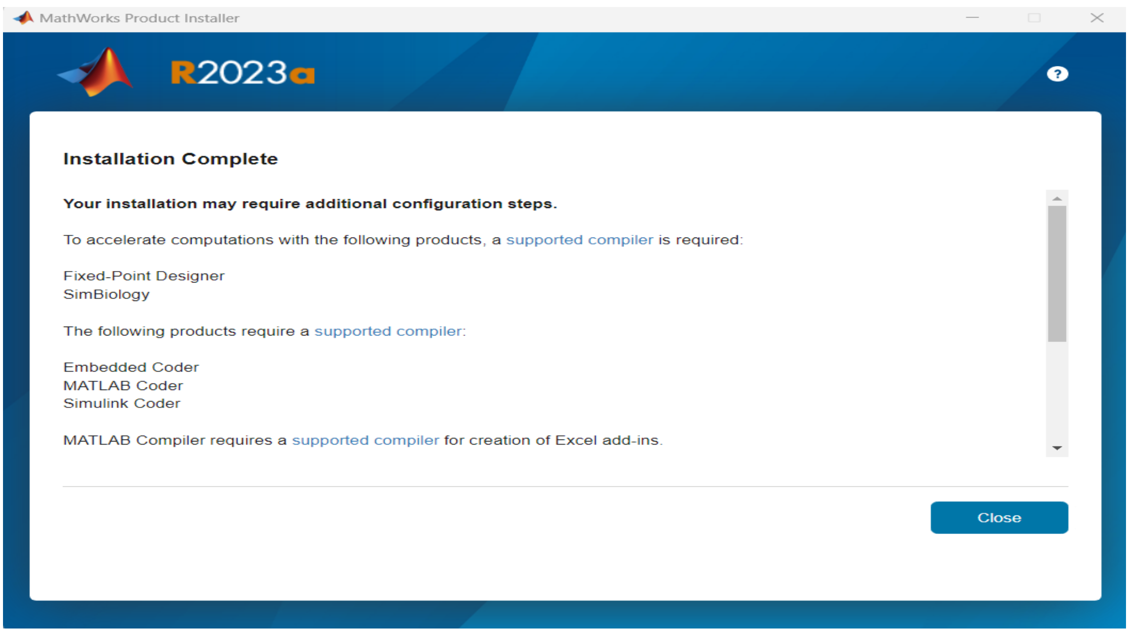


**Figure 5: Installation Complete**

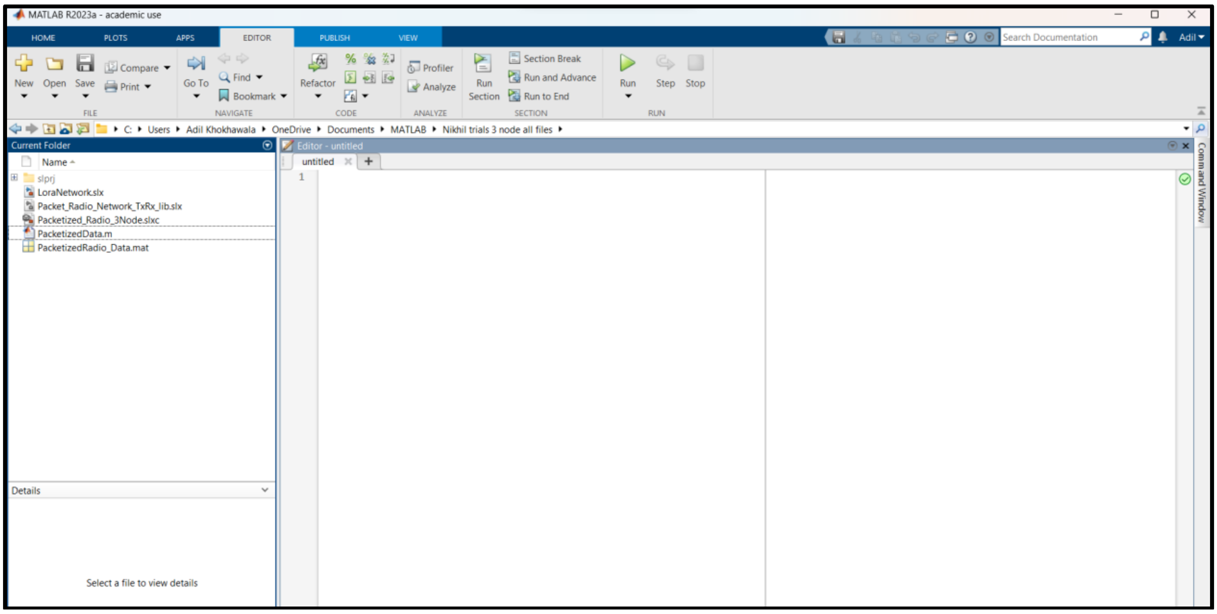


**Figure 6: MATLAB boot-up interface**

# 3 Running the simulation:

Now that the simulation environment is set up, we run the files in a specific order so that the Simulink files have received the node signals, which are running and can generate network traffic.

**Step 1:** The zip file needs to be extracted and copied to the MATLAB folder path. By doing this, we can then add the code files to the path for execution.

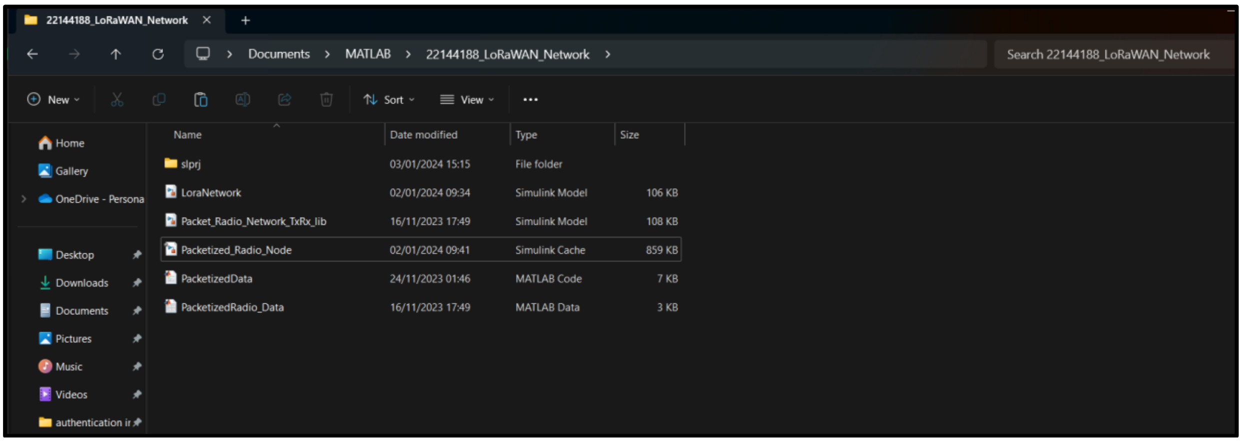


**Figure 7: Files saved in the correct path**

**Step 2:** For generating the LoRa Beacon signals, the files Lora Tx and Lora Rx are run. These are the files which contain the parameters inside the transmitter and receiver. After which, the Pilot.m file is run, to generate these beacon signals. They are primarily based on factors like spreading factor, bandwidth, power, frequency channel etc. Once this is run, a payload is created which uses the message fed as input. The IQ signal is passed on to the Lora receiver file with the other parameters to retrieve the message sent by the transmitter.

This process showcases the signal processing method in the LoRaWAN protocol in accordance with the LoRa Alliance.

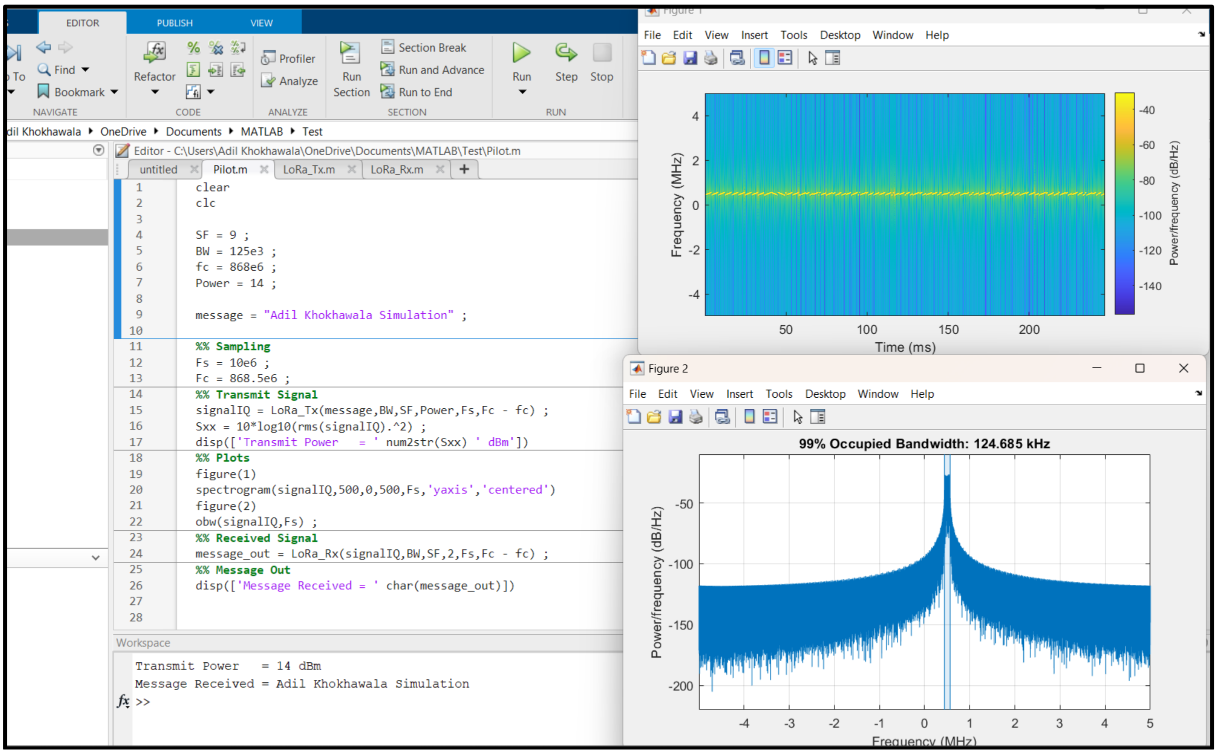


**Figure 8: LoRa Beacon Simulation**

Factors selected :

- ✓ Spreading Factor (SF) : 9
- ✓ Bandwidth (BW) : 125 KHz
- ✓ Centre Frequency: 868 MHz
- ✓ Power: 14 dBm
- ✓ Message : "Adil Khokhawala Simulation"

**Step 3:** To run the Simulink model for data packet processing, we need to run the Node frame file labelled (***PacketizedData.m***). Upon running this file, the requirements for 10 nodes are sent to the Simulink model where it is being called.

The ‘*Packetizeddata.m’* file contains the transmitter and receiver codes where the nodes relate to a node 11, i.e. a gateway.

Editor - PacketizedData10Node.m | Scopes - Rx Signal Power

PacketizedData10Node.m

```
clear

% Load AGC and SYNC symbols.  They are taken from data sequence "AGC_SYNC_Data"
load AGC_SYNC_Data;  % Load AGC bits, SYNC bits, and st, the QPSK symbol string for DFE training. The training string
                     % is the QPSK symbols of the bit string of AGC and
                     % SYNC.  The modulation rate is 2 bits per symbol.

% Generate Data Frame TxData for all 11 Ndoes (i=1,2,3)
for i=1:11

TxData.Agc=ppdu_bb(1:50,1);  %Load AGC signal, 2 bits per symbol
TxData.Sync=ppdu_bb(51:304); %Load Sync Signal, 2 bits per symbol

N=(130)*8;
PyldLengthD=4+2+2+8+8+8+32+N+32;                    % PayLoadBitsSize= 4+2+2+8+8+8+32+N+32 = 1136 @N=130*8.
PyldLength=decimalToBinaryVector(PyldLengthD, 16);  % Convert a decimal number to a binary vector

% Add CRC Checksum Using Matlab CRC Generator
crcGen=comm.CRCGenerator();
x=logical(PyldLength)';      %CRC-16 with 16 Checksum
plcrc=crcGen(x);
TxData.PayloadLength=PyldLength';  %Load Payload Length
TxData.HeaderCrc=plcrc(17:32,1);   %Load Header CRC

% Generate the MAC Header
% | Type 4b | Version 2b | Reserved 2b | To Add 8b | From Add 8b |
%  Sequence No. 8b| Time Stamp 32b |
```

**Figure 9: Creating frame for packets**

Editor - PacketizedData10Node.m | Scopes - Rx Signal Power

PacketizedData10Node.m

```
rng(11);
TxData.Payload=randi([0 1], [N 1]);
TxData.Payload(length(TxData.Payload),1)=0;
PyldBits=[ TxData.Type; ...
           TxData.Version; ...
           TxData.Reserved; ...
           TxData.ToAddress; ...
           TxData.FromAddress; ...
           TxData.SequenceNo; ...
           TxData.Time; ...
           TxData.Payload];

crcGen=comm.CRCGenerator('z^32+z^26+z^23+z^22+z^16+z^12+z^11+z^10+z^8+z^7+z^5+z^4+z^2+z^1+1');
reset(crcGen);

PyldCRC=crcGen(PyldBits);

% Final TxData Frame is
TxData.PayloadCRC=PyldCRC(length(PyldCRC)-31:length(PyldCRC),1);
TxSignalBits= [TxData.Agc; ...
              TxData.Sync; ...
              TxData.PayloadLength; ...
              TxData.HeaderCrc; ...
              TxData.Type; ...
              TxData.Version; ...
              TxData.Reserved; ...
              TxData.ToAddress; ...
              TxData.FromAddress; ...
              TxData.SequenceNo; ...
              TxData.Time; ...
              TxData.Payload; ...
              TxData.PayloadCRC];
    if i==1
    TxDataBitsN1=TxSignalBits;        % Rename the Data Frame for Node1
```

**Figure 10: Tx_Data Frame Creation**

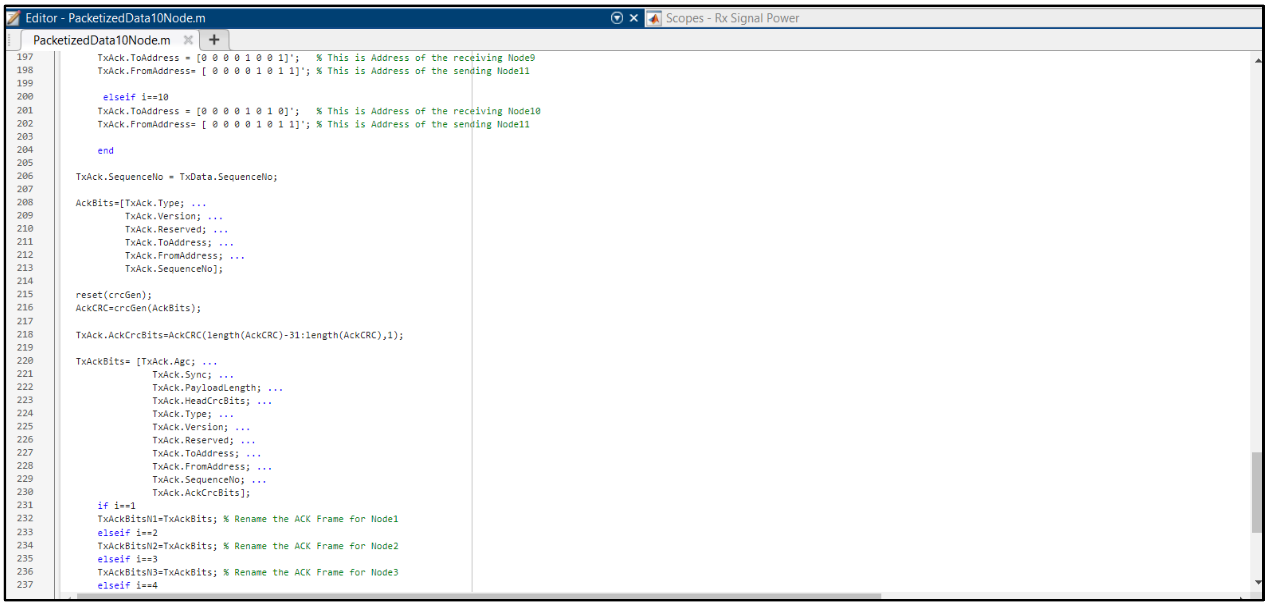


**Figure 11: Tx_Ack Frame Creation**

**Step 4:** We must now run the MALTAB data file labelled *"Packetized+Data.mat"* , this file defines all the variables present in the code. Factors like Ts, MDSN, Node definition, bandwidth etc. are defined. These can be run or either loaded from the **".mat"** file.

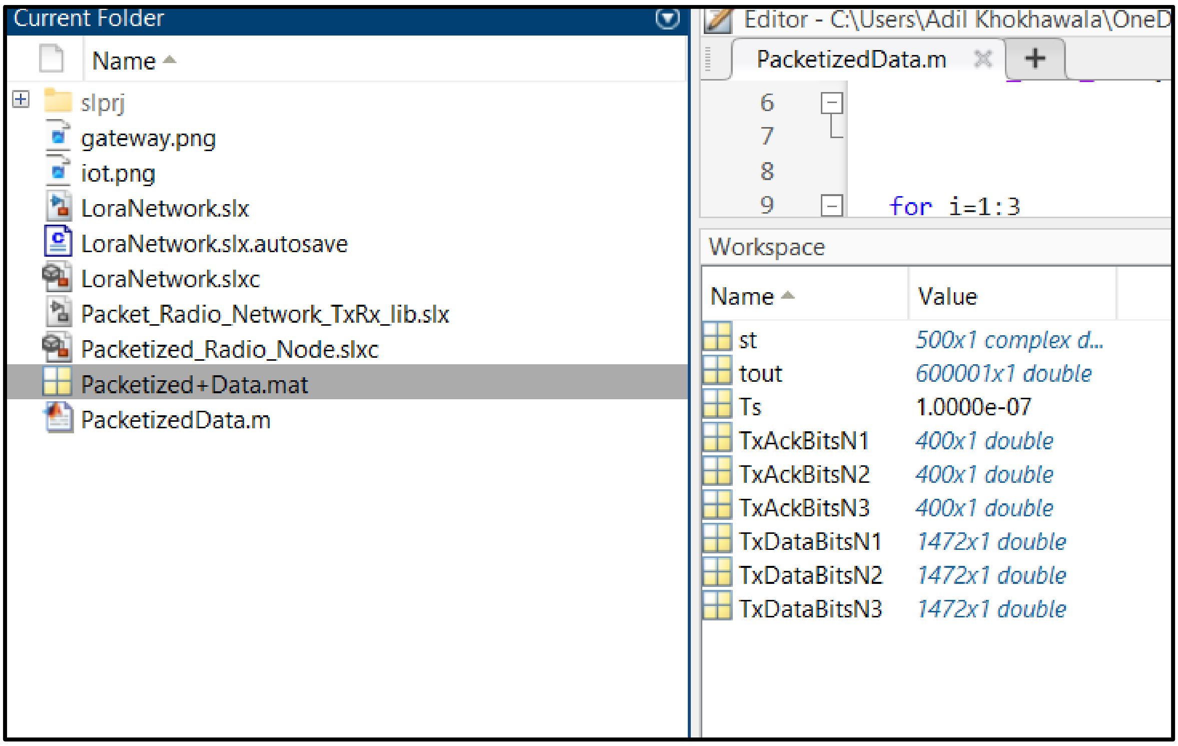


**Figure 12: MATLAB data file**

**Step 4:** Next step is to run the Simulink file ***LoraNetwork.slx***. This runs the created Simulink model in which the network is defined and connected to the gateway in a star topology. Additionally, each node is defined with a transmitter and receiver through Rayleigh SISO channels and AWGN Channel to generate signal. We have used these 2 channels as an alternate because MATLAB does not support lora blocks in Simulink.

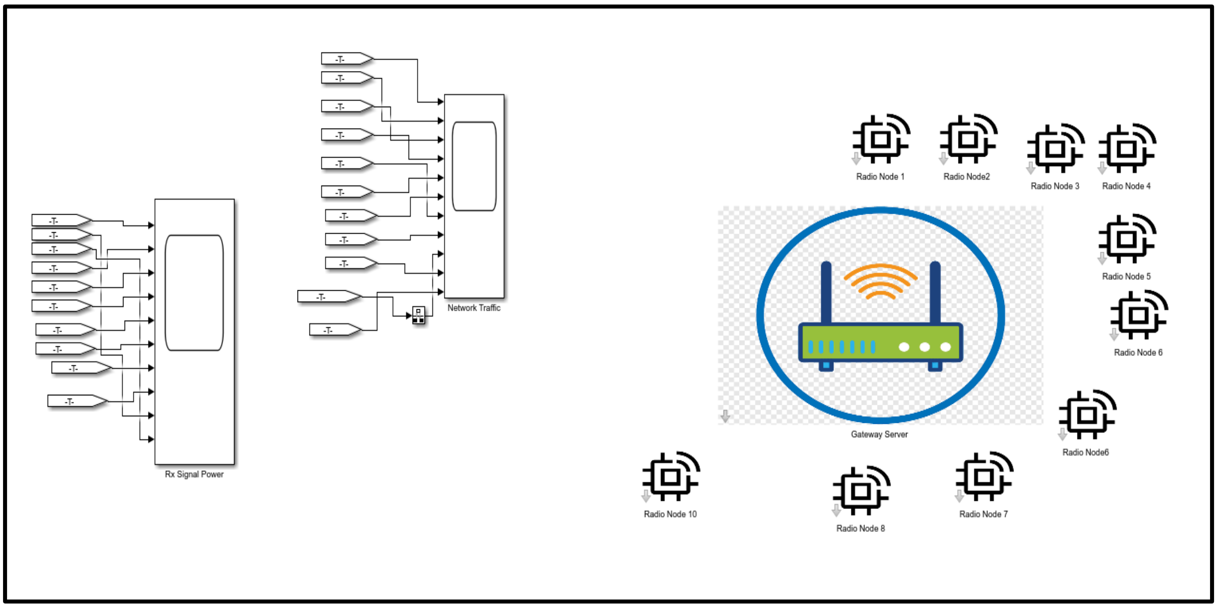


**Figure 13: LoRa Network with 10 Nodes**

Once the files are run in the order mentioned above the Lora network starts to produce network traffic, i.e. a bidirectional communication takes place between the sensor nodes and the gateway server.


**Figure 14: Transmitted Signals**

The below image shows the received power at the sensor node end.

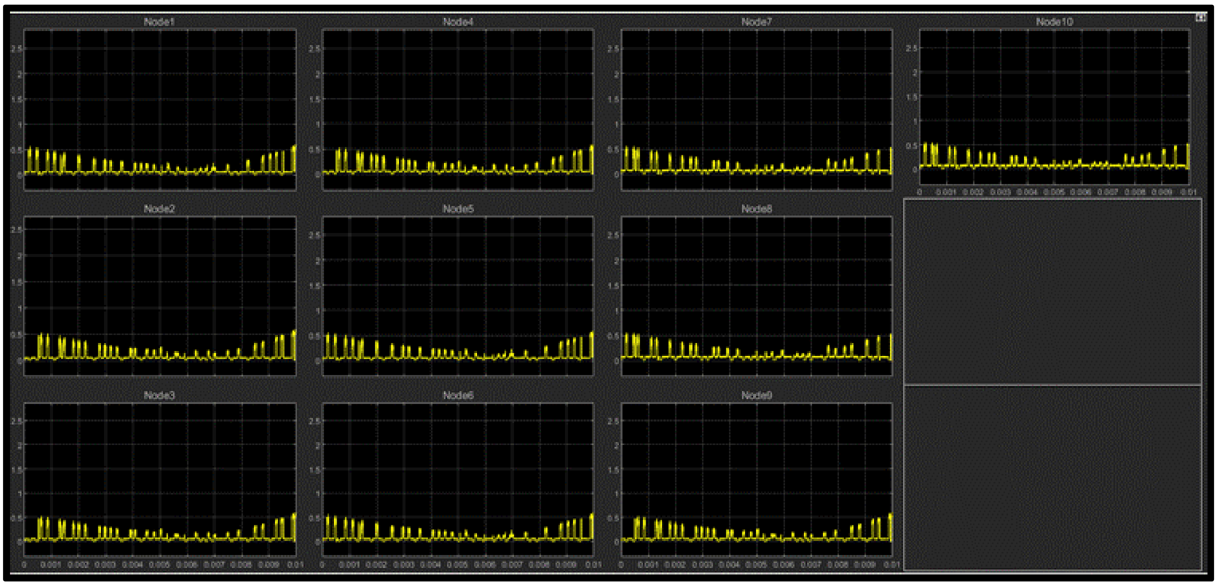


**Figure 15: Received Signal Power (Rx)**

# 4 Poc Implementation

The Poc of the proposed solution was implemented by the company's research team. The code developed for the time-synchronized random see generator and the code for checking the authenticity at the gateway end were provided to the company. They flashed the code onto the ESP32 Chip and the Prasgate LoRa node. This is a general-purpose chip which is placed before the gateway server. This research does not alter any rule/standard in the LoRa protocol; it only adds an additional layer of security to increase the complexity and time penalty for any attacker that is trying to impersonate a compliant node in the network. Our solution checks the parameter and the random seed value at both ends and matches the timestamp. If the timestamp is less than 5 minutes, it allows the packet to pass through to the application server, else it discards the packet and throws a message stating, **"Rogue Device, Invalid Packet!".**

Below is an example of how the random seed value is appended to the authentication packet, which increase the complexity of the entire data transmission process and makes it difficult for a threat actor to intercept the packet and disrupt the network.

| | |
|---|---|
| <nonce value> | b'53027dz12c5oJzqRLMrRU0mRz3128930dd3f52e3ea' |

**Figure 6 : Authentication packet**

| |
|---|
| 91b'53027dz12c5oJzqRLMrRU0mRz3128930dd3f52e3ea' |

**Figure 7: Appended packet sent for further transmission.**

Below attached are Poc code snippets provided to the company's research team for flashing onto the ESP32 Chip and the Prasgate Lora Node to implement the time-synchronous random seed generator. The code was modified at the company's end for implementing it on the hardware level.

```c
#include <stdio.h>
#include <stdlib.h>
#include <time.h>

int generate_random_number() {
    return rand() % 100 + 1;
}

void append_random_number(char *message) {
    int random_number = generate_random_number();
    sprintf(message + strlen(message), " %d", random_number);
}

void append_timestamp(char *message) {
    time_t rawtime;
    struct tm *timeinfo;

    time(&rawtime);
    timeinfo = localtime(&rawtime);

    char buffer[20];
    strftime(buffer, sizeof(buffer), " %Y-%m-%d %H:%M:%S", timeinfo);
    strcat(message, buffer);
}

int main() {
    char input_message[1000];
    printf("Enter your message: ");
    fgets(input_message, sizeof(input_message), stdin);
    input_message[strcspn(input_message, "\n")] = '\0';

    char composite_output[1000];
    strcpy(composite_output, input_message);
    append_random_number(composite_output);
    append_timestamp(composite_output);
    printf("%s\n", composite_output);

    return 0;
}
```

**Figure 16: Random seed appended to data packet**

```c
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <string.h>

int generate_random_number() {
    return rand() % 100 + 1;
}

void append_random_number(char *message) {
    int random_number = generate_random_number();
    sprintf(message + strlen(message), " %d", random_number);
}

void append_timestamp(char *message) {
    time_t rawtime;
    struct tm *timeinfo;

    time(&rawtime);
    timeinfo = localtime(&rawtime);

    char buffer[20];
    strftime(buffer, sizeof(buffer), " %Y-%m-%d %H:%M:%S", timeinfo);
    strcat(message, buffer);
}

int is_valid_message(char *composite_output) {
    int random_number_recovered, random_number_generated;
    char timestamp[20];

    sscanf(composite_output, "%*[^0-9]%d", &random_number_recovered);
    strcpy(timestamp, strrchr(composite_output, ' ') + 1);

    random_number_generated = generate_random_number();

    if (random_number_recovered != random_number_generated) {
        return 0;
    }

    time_t current_time, message_time;
    struct tm tm;
```

**Figure 17: Gateway checking the authenticity**

```c
    time(&current_time);
    sscanf(timestamp, "%d-%d-%d %d:%d:%d",
           &tm.tm_year, &tm.tm_mon, &tm.tm_mday,
           &tm.tm_hour, &tm.tm_min, &tm.tm_sec);
    tm.tm_isdst = -1;
    message_time = mktime(&tm);

    long diff = current_time - message_time;
    if (diff < 0) {
        diff = -diff;
    }

    return diff < 5 * 60; // if the difference between timestamp is less than 5 mins.
}

int main() {
    char input_message[1000];
    printf("Enter your message: ");
    fgets(input_message, sizeof(input_message), stdin);
    input_message[strcspn(input_message, "\n")] = '\0';

    char composite_output[1000];
    strcpy(composite_output, input_message);
    append_random_number(composite_output);
    append_timestamp(composite_output);

    if (is_valid_message(composite_output)) {
        printf("Valid packet: %s\n", composite_output);
    } else {
        printf("Rogue Device, Invalid packet: %s\n", composite_output);
    }

    return 0;
}
```

**Figure 18: Gateway checking the authenticity**

# 5 Conclusion

The configuration manual contains the detailed steps to execute the implementation process of the research project in a sequential format and makes it easy for any user to understand the flow of the model.

# 6 Monthly Internship Activity Report

The internship commenced on 25th September 2023 to 10th December 2023.

## Monthly Internship Activity Report

The Internship Activity Report is a 1-page monthly summary of the activities performed by you andwhat you have learned during that month. The Internship Activity Report must be signed off by your Company and included in the configuration manual as part of the portfolio submission.

Student Name: **Adil Mustafa Khokhawala**

Student number: **22144188**

Company: **Tynatech Ingeneous Pvt. Ltd**.

Month Commencing: **October 2023**

1. Introduction to company catalog
2. Introductory interaction with Solution team for
   a. Water Management solution.
   b. Tynarace Solutions
   c. Health Monitoring and Energy monitoring
3. Introduction to LoRaWAN technology

4. Research Work
   a. Create a deck to summarize two papers on vulnerabilities in LoRaWAN
   Create a deck to summarize the security vulnerabilities in LoRaWAN network.
   b. Create a deck for authentication methods, security issues caused due to lack of authentication in LoRaWAN.
5. Work with the ticketing team and engage in troubleshooting calls.

Employer comments

Adil puts a lot of effort into his work. He has demonstrated a good grasp of topics and has a positive attitude while learning and approaching tasks. He has been engaging very well in his day-to-day activities and in his research topic as well.

Student Signature: ______________________ Date: 01/11/2023

Industry Supervisor Signature: ______________ Date: 01/11/2023

# Monthly Internship Activity Report

The Internship Activity Report is a 1-page monthly summary of the activities performed by you and what you have learned during that month. The Internship Activity Report must be signed off by your Company and included in the configuration manual as part of the portfolio submission.

Student Name: **Adil Mustafa Khokhawala**

Student number: **22144188**

Company: **Tynatech Ingeneous Pvt. Ltd**.

Month Commencing: **November 2023**

1. Configuration of LoRaWAN Chip, Chip no. PRASGATE LORA node with LoRa Chip SX1276 & SX1278.
2. Hands-on configuration for the current solutions implemented in the company.
   a. Water Management solution.
   b. Tynarace Solutions
   c. Health Monitoring and Energy monitoring
3. Research Work
   a. Simulation environment setup in MATLAB
   b. Create a working node-to-node connection in LoRaWAN
   c. Implement the working LoRaWAN network with at least 6 nodes
   d. Simulate improvements in the authentication mechanism

5. Coordinate with operations team on repetitive problems arising in the tickets, engage in troubleshooting calls.

Employer comments

Adil always takes an initiative to learn. He has a positive attitude towards any task assigned to him and engages himself in KT sessions to enhance his learning. The research and Operations team were happy with his performance.

Student Signature: __________ Date: 30/11/2023

Industry Supervisor Signature: __________ Date: 30/11/2023