Group 9 – phase 3 – Testing and reporting of the IT environment

Table of Contents

[Project info 1](#_Toc93421190)

[Test objective 4](#_Toc93421191)

[Test summary 7](#_Toc93421192)

[Defects 7](#_Toc93421193)

[Resources: 7](#_Toc93421194)

# Project info

Phase 2 of the project is based on researching LoRa devices and the way they communicate sensor data. The main research question is **what are the options, alternatives, and challenges to monitor the security of such an IoT application?**

A LoRa sensor network communicates sensor data via a gateway to a cloud environment, to be used in applications. The goal of the project was to create an application that monitors this data, filters it, and then sends it to its destination. We have achieved this by working on Node-red to create the user interface and the filtering options, along with a LoRa device and work on The Things Network website.

* **Node-red application**

The project team decided to work on the user interface and application functionality on the Node-red platform.

Node-red is a platform made to simplify building apps for low-end devices.  
By the usage of connecting nodes in an order, each performing a single step in the process of the application with pre-build and optimized code for low-end devices, node-red allows for the developers to write efficient code more easily for low-end devices. Since the app will be running on a raspberry pi 3B performing as gateway as well, the project team decided to not build a full-stack application with many features. Node-red helped the project team to keep the application simple and efficient, prioritizing the raspberry pi 3B on being an efficient gateway instead of a host for the application.

Below there are some pictures of the node-red application building process using nodes connected and ordered through wires between the nodes.

The entire Node-red configuration:Diagram

Description automatically generated

The key generation configuration (the first picture) and UI (the next 2 pictures): There is a specific node for each button (generate key and save generated key), label (generated key), and list (key type). The DevAddr node provides the items of the list and specifies the default one (DevAddr): Diagram, timeline

Description automatically generatedDiagram

Description automatically generatedGraphical user interface, application

Description automatically generatedGraphical user interface, text

Description automatically generated

The table and table configuration:Afbeelding met tekst, schermafbeelding, binnen, monitor

Automatisch gegenereerde beschrijving

The log related functions take the received messages, read them, convert them to string to be displayed in the table and store them in the log file. Log entry notifier is the pop-up message that is being displayed every time a new log entry appears. Newest log is a label that displayed the latest entry right above the table. Log list table contains the code for the table: its aspect and elements.Diagram

Description automatically generated

# Test objective

* **UI testing**

The user interface of this application is not built like a traditional website, but instead there have to be nodes for each button, label, table, etc. the layout of the page is not determined in a CSS file, for example, but in this interactive menu below:Chart, bar chart

Description automatically generated

The table contains its own separate node that contains its style and content. The label is also a separate node:A screenshot of a computer

Description automatically generated with medium confidence

* **Node-red testing**

Given that the work on the node-red application has been done by members of the team who are studying software engineering, they approached the testing the usual software way, creating test cases and analyzing each scenario. This is the Node-red alternative to unit testing. Instead of the usual test code and syntax, new testing nodes has been created for each scenario. This has been helpful from the beginning, given that only a very specific set of messages should be displayed and stored. Most Node-red tests have been json test data nodes, but the udp simulator node present in the screenshot below is also a test node. Before connecting the application to the LoRa device, the students have experimented with simulated data coming from the test nodes.Diagram

Description automatically generated with medium confidenceA screenshot of a computer

Description automatically generated with low confidence

A picture containing text, shoji, indoor, kitchen appliance

Description automatically generated

Some of the test nodes provide wrong or incomplete data on purpose to test the response of the application. The output is different depending on what the node receives. In the first example, there is nothing written in the message field. The second example shows that the rssi field is missing (undefined). A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generatedSimulated decrypted message (picture above) data: {"in":"","out":"496e20726561636821","buffers":{}}

Sending a partially decrypted data message: �is �'������{"rxpk":[{"tmst":167888406,"chan":0,"rfch":0,"freq":868.100000,"stat":1,"modu":"LORA","datr":"SF9BW125","codr":"4/5","lsnr":13,"rssi":-78,"size":22,"data":"QEILJhgAAX"}]} results in an error:

Graphical user interface, text, application

Description automatically generated

If the message is sent without the data property: �is �'������{"rxpk":[{"tmst":167888406,"chan":0,"rfch":0,"freq":868.100000,"stat":1,"modu":"LORA","datr":"SF9BW125","codr":"4/5","lsnr":13,"rssi":-78,"size":22}]}, the message will be ignored by the application and the other content related to it will not be displayed:Graphical user interface, application

Description automatically generated

If the message is sent with rxpk property as empty array: �is �'������{"rxpk":[]}, it will be ignored by the application and not displayed on the UI table:Graphical user interface, text, application

Description automatically generated

If a random string is inputted in the UDP field, the following error will be displayedGraphical user interface, text, application

Description automatically generated

A random non string message (1234532) results in caught error: Graphical user interface, text, application

Description automatically generated

Web hacking techniques such as SQL injection or Cross-site scripting are not possible because there is no written user input in the application.

After performing the Node-red application testing, the solution that the team has come with is to create a security check in the application. Every incoming message is analyzed and if it fits 100% with the default criteria, it will be displayed on the user interface table. Otherwise, the message will be ignored. A corrupted message will not affect the application in anyway, so it will not crash.

Another early test is about the timestamp of the valid messages. If a second message is sent before a certain interval of time, it will not count as valid, and it will be listed as duplicate timestamp.

Also, a default value for the keys needs to be put in place before any generation in order to avoid errors. To fix this, a node has been created that chooses DevAddr as the default key option and the last key of that type that has been generated. This way, the key label is never empty.

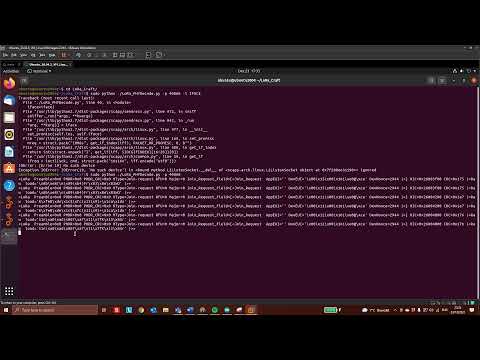
Early node-red testing with test data (the video has been recorded before Christmas):[](https://www.youtube.com/embed/YxYxUOFx7F0?feature=oembed)

* **Hardware testing**

For hardware testing we first started off by erasing any data from our devices that was left over from previous projects, like our SD card. Then, we simply tried to understand the LoRa protocol and see if our sending devices were working by simply connecting them and seeing if they could receive and send (full duplex). The next step was to connect a sending device to our gateway. To do this, we first uploaded our filesystem to the SD card. We went over each pin of our raspberry pi and shield to see if any of them were broken or malfunctioning. Then, we connected everything together. We then tested if we could send data from our sending device through our gateway to the Things Network. Once that worked we started to connect our sending device through the gateway to our own application. We then used the hackRF to see if we could intercept any useful data. However, we weren’t able to, but learned more about the protocol in the end and got a better grasp of the vulnerabilities of our setup.

Demonstration and explanation about hardware testing will be part of the Friday project presentation.

Explanation about HackRF: testing during the project (the video was recorded before Christmas)

[](https://www.youtube.com/embed/4kO-bzaRzd4?feature=oembed)

# Test summary

The testing process has revealed a number of issues with the message analysis process, the main one being related to displaying the wrong message in the debugger. Throughout many tests, if anything was wrong with the sent data, the message will be shown as undefined, even if that parameter was correct. This did not affect the flow of the application but was a potential weak spot in our application’s security. Given the limited capabilities of the Node-red application and limited scope, the number of unit tests was relatively low, 12 tests having been made to test the application’s response to received messages. The application has this only feature, so this was the only one that could have been subject to testing. Once all the tests passed, the application became ready for connection with the LoRa device.

# Defects

The main bug found during testing was the timestamp display in the user interface table. In the first few versions of the Node-red code, it would not be displayed correctly, but this has been solved quite quickly. Another initial bug was the display of the incomplete messages in the debugger. If anything was incomplete or wrong in the payload, the message field would be listed as undefined, but that has also been solved. All the known bugs of the application have been closed.

# Resources:

* <https://geteasyqa.com/qa/write-test-report/>
* <https://www.tricentis.com/blog/64-essential-testing-metrics-for-measuring-quality-assurance-success/>
* <https://www.thethingsnetwork.org/>