AIR QUALITY MONITORING

USING IOT

INTRODUCTION:

- As the name suggests, embedded describes something that is connected to another item. An embedded system is a piece of computer hardware that also contains software. A stand-alone unit or a component of a bigger system can both be an embedded system. An embedded system with a microcontroller or microprocessor is designed to carry out a certain purpose. For instance, a fire alarm is an integrated device that only detects smoke. An embedded system is similar to a computer system in that it is primarily designed to accomplish specific functions such as controlling data in various electronics-based systems, accessing data, processing, and storing data. Embedded systems are hardware and software combinations that are designed to perform a certain set of functions. The embedded system's most essential feature is that it regenerates the output in a very short amount of time. Many embedded systems will be encountered in our daily lives.OCK DIAGRAM:

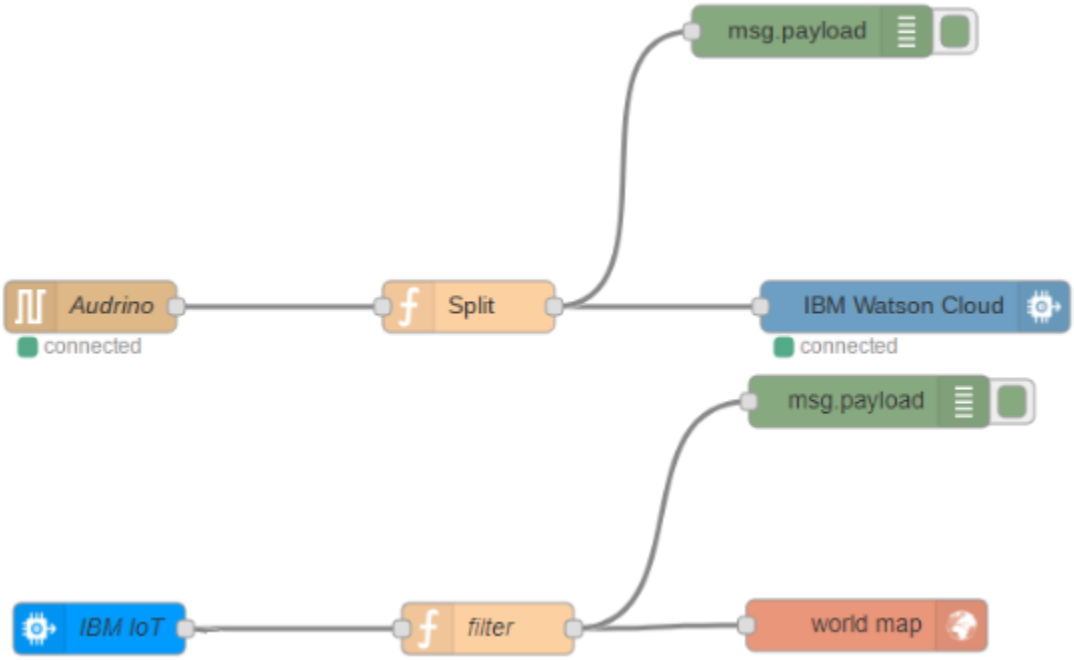
Node-RED Vulnerabilities:

Node-RED is “a programming tool for wiring together hardware devices, APIs and online services”, which provides a way of “low- code programming for event- driven applications” [36]. As an open-source platform, Node-RED is mainly tar- geted for deployment as a single-user platform, although it is also available on the IBM Cloud platform [23].

Node-RED platform:

A node is a reactive Node.js application triggered by receiving messages on at most one input port (dubbed source) and sending the results of (side-effectful) computations on output ports (dubbedsinks), which can be potentially multiple, unlike the input port. Figure 3 illustrates the code structure of a Node-REDnode. A special type of node without sources and sinks, called configuration node, is used for sharing configuration data, such as login credentials, between multiple nodes. A flow is a representation of nodes connected together. End users can either create their own flows on the platform’s environment or deploy existing flows pro- vided by the official Node-RED catalog [33] and by third parties. In Node-RED, contexts provide a shared

communication channel between different nodes without using the explicit messages that pass through a flow [40]. Therefore the node wiring visible in the user interface reflects only a part of the information flows that are possible in the flow. It introduces an implicit channel that is not visible to the user via the graphical interface of a flow. Node-RED defines three scope levels for the contexts



The provided policies can later be vetted by the platformand the user, before deploying the node. SandTrap [3] offers a pol- icy generation mechanism to aid developers in designing the policies, enabling both baseline and advanced policies customized by developers or users to express fine-grained app- specific security goals. In the following, we discuss Node-RED attacks and vulnerabilities that mo- tivate enriching the policy mechanismwith different granularity levels. These policies will further be formalized in Section 3.

Platform-level isolation vulnerabilities:

All APIs provided by the underlying runtimes, Node-RED and ⟨ ⟩ Node.js, are accessible for node developers, as well as the incom- ing messages within a flow. As shown in Figure 6a, there are various attack sce- narios for malicious nodes [3]. At the Node.js level, an attacker can create a ma- licious Node-RED node including.

Program:

IOT.py - C:\Users\flower\AppData\Local\Programs\Python\Python311\IOT.py (3.11.4)

File Edit Format Run Options Window Help

```
from tkinter import *
import requests
from bs4 import BeautifulSoup
# link for extract html data
def getdata(url):
    r = requests.get(url)
    return r.text
def airinfo():
    soup = BeautifulSoup(htmldata, 'html.parser')
    res_data = soup.find(class_="DonutChart--innerValue--2rO41 AirQuality--extendedDialText--2AsJa").text
    air_data = soup.find_all(class_="DonutChart--innerValue--2rO41 AirQuality--pollutantDialText--3Y7DJ")
    air_data=[data.text for data in air_data]
    ar.set(res_data)
    o3.set(air_data[0])
    no2.set(air_data[1])
    so2.set(air_data[2])
    pm.set(air_data[3])
    pml.set(air_data[4])
    co.set(air_data[5])
    res = int(res_data)
    if res <= 50:
        remark = "Good"
        impact = "Minimal impact"
    elif res <= 100 and res > 51:
        remark = "Satisfactory"
        impact = "Minor breathing discomfort to sensitive people"
    elif res <= 200 and res >= 101:
        remark = "Moderate"
        impact = "Breathing discomfort to the people with lungs, asthma and heart diseases"
    elif res <= 400 and res >= 201:
        remark = "Very Poor"
        impact = "Breathing discomfort to most people on prolonged exposure"
    elif res <= 500 and res >= 401:
        remark = "Severe"
        impact = "Affects healthy people and seriously impacts those with existing diseases"
    res_remark.set(remark)
    res_imp.set(impact)
# object of tkinter
# and background set to grey
master = Tk()
master.configure(bg='light grey')
```

Ln: 1 Col: 0

IOT.py - C:\Users\flower\AppData\Local\Programs\Python\Python311\IOT.py (3.11.4)

File Edit Format Run Options Window Help

```
# Variable Classes in tkinter
air_data = StringVar()
ar = StringVar()
o3 = StringVar()
no2 = StringVar()
so2 = StringVar()
pm = StringVar()
pml = StringVar()
co = StringVar()
res_remark = StringVar()
res_imp = StringVar()
# Creating label for each information
# name using widget Label
Label(master, text="Air Quality : ",
      bg="light grey").grid(row=0, sticky=W)
Label(master, text="O3 (µg/m3) :",
      bg="light grey").grid(row=1, sticky=W)
Label(master, text="NO2 (µg/m3) :",
      bg="light grey").grid(row=2, sticky=W)
Label(master, text="SO2 (µg/m3) :",
      bg="light grey").grid(row=3, sticky=W)
Label(master, text="PM2.5 (µg/m3) :",
      bg="light grey").grid(row=4, sticky=W)
Label(master, text="PM10 (µg/m3) :",
      bg="light grey").grid(row=5, sticky=W)
Label(master, text="CO (µg/m3) :",
      bg="light grey").grid(row=6, sticky=W)
Label(master, text="Remark :",
      bg="light grey").grid(row=7, sticky=W)
Label(master, text="Possible Health Impacts :",
      bg="light grey").grid(row=8, sticky=W)
# Creating label for class variable
# name using widget Entry
Label(master, text="", textvariable=ar,
      bg="light grey").grid(
    row=0, column=1, sticky=W)
Label(master, text="", textvariable=o3,
      bg="light grey").grid(
    row=1, column=1, sticky=W)
Label(master, text="", textvariable=no2,
      bg="light grey").grid(
```

Ln: 1 Col: 0

IOT.py - C:\Users\flower\AppData\Local\Programs\Python\Python311\IOT.py (3.11.4)

File Edit Format Run Options Window Help

```
      bg="light grey").grid(row=5, sticky=W)
Label(master, text="CO (µg/m3) :",
      bg="light grey").grid(row=6, sticky=W)
Label(master, text="Remark :",
      bg="light grey").grid(row=7, sticky=W)
Label(master, text="Possible Health Impacts :",
      bg="light grey").grid(row=8, sticky=W)
# Creating label for class variable
# name using widget Entry
Label(master, text="", textvariable=ar,
      bg="light grey").grid(
    row=0, column=1, sticky=W)
Label(master, text="", textvariable=o3,
      bg="light grey").grid(
    row=1, column=1, sticky=W)
Label(master, text="", textvariable=no2,
      bg="light grey").grid(
    row=2, column=1, sticky=W)
Label(master, text="", textvariable=so2,
      bg="light grey").grid(
    row=3, column=1, sticky=W)
Label(master, text="", textvariable=pm,
      bg="light grey").grid(
    row=4, column=1, sticky=W)
Label(master, text="", textvariable=pm1,
      bg="light grey").grid(
    row=5, column=1, sticky=W)
Label(master, text="", textvariable=co,
      bg="light grey").grid(
    row=6, column=1, sticky=W)
Label(master, text="", textvariable=res_remark,
      bg="light grey").grid(row=7, column=1, sticky=W)
Label(master, text="", textvariable=res_imp,
      bg="light grey").grid(row=8, column=1, sticky=W)
# creating a button using the widget
b = Button(master, text="Check",
           command=airinfo, bg="Blue")
b.grid(row=0, column=2, columnspan=2,
       rowspan=2, padx=5, pady=5,)
mainloop()
```

Ln: 1 Col: 0

OUTPUT:

Air Quality :	85
O3 (µg/m3) :	67
NO2 (µg/m3) :	22
SO2 (µg/m3) :	13
PM2.5 (µg/m3) :	30
PM10 (µg/m3) :	45
CO (µg/m3) :	479
Remark :	Satisfactory
Possible Health Impacts :	Minor breathing discomfort to sensitive people