The SimTech Group

Deliverable H – Prototype II & Customer Feedback

GNG1103 (A00) – Engineering Design

Fall 2024

University of Ottawa

Group #17

Nour Mokdad

Pavithra Raj Mohan

Kushal Raveen Jayarathna

Stéphane Lauzon-Brisson

1 – Introduction

In this document, we complete our final prototype. This prototype will be a comprehensive prototype combining our previous prototypes with the addition of the Quality of Service (QoS) Reduction using the QoS Bandwidth API from Shabodi.

2 – Prototype III: Test Plan, analysis and results

The Third prototype has as its goal the combination of our previous work with a request to the QoS Bandwidth API from Shabodi. After simulating a SIM Swap, the application returns the location, Device ID and time that the event occurred. The application also reduces the device's bandwidth using the QoS reduction API. The user has the ability to reinstate the QoS or keep it reduced by entering a command in the terminal.

2.1 Terminal Responses

When the SIM swap simulation is initiated by entering "SWAP", the application takes a snapshot of the Device ID, time and location (entered manually) of the incident. When the administrator accepts the change in SIM by entering "y" in the terminal, the application calls the QoS Bandwidth API a second time to reinstate the QoS.

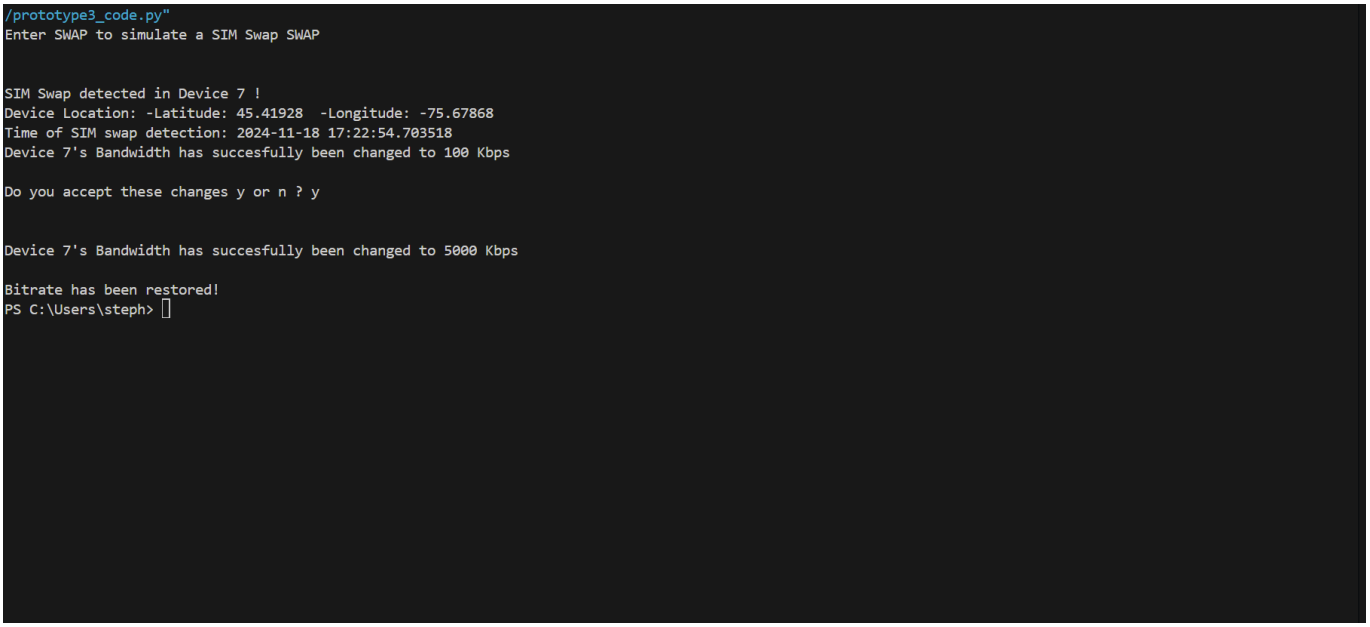

Figure 1. Response to accepting the SIM Swap

If the administrator does not accept the change, in this case by entering “n” in the terminal, the application does not reinstate the QoS, and the administrator is asked to alert the necessary people.

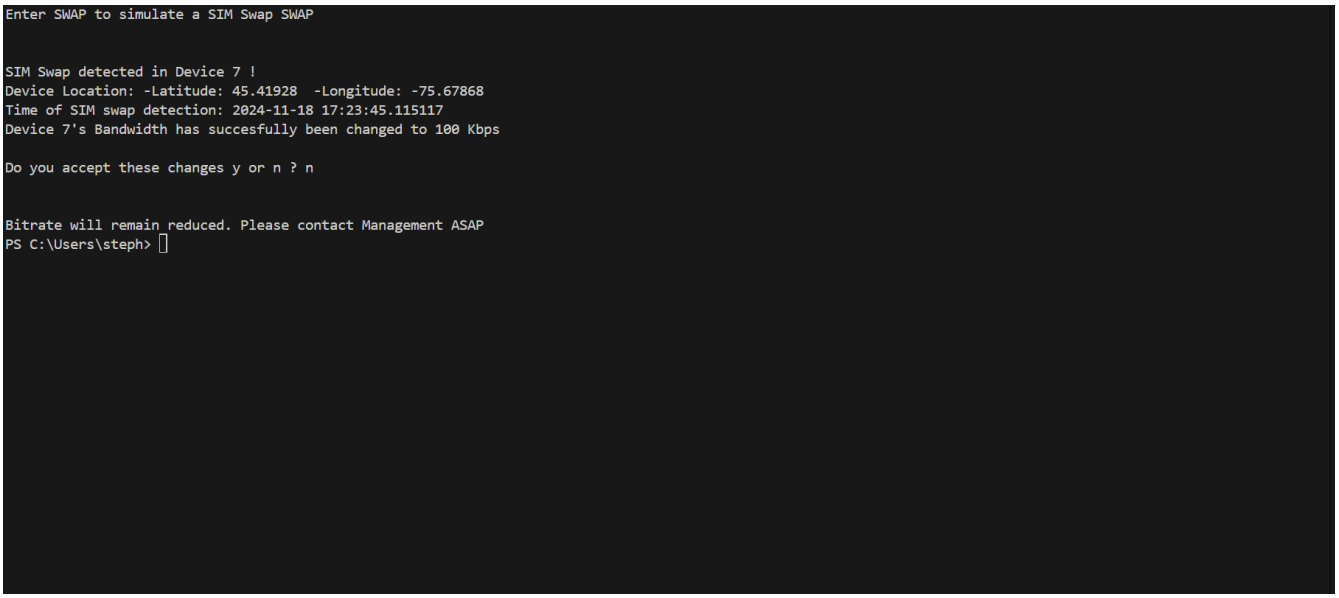

Figure 2. Response to accepting the SIM Swap

2.2 Testing of prototype 3

The code seen in Figures 3-6 was run 20 separate times with the settings seen in Figure 3. Ten of the tests accepted the SIM Swap whereas 10 others denied the SIM Swap. The initial Bitrate reduction is deemed "successful" if the API does not return an error code. The Python code is set up to print that there was an error in this case. The responses to the tests are outlined in Table 1 below.

Table 1. Tests run for Prototype 3

Test #	Initial Bitrate reduction successful ? (Y or N)	Swap Accepted ? (Y or N)	Response as expected? (Y or N)
1	Y	Y	Y
2	Y	Y	Y
3	Y	Y	Y
4	Y	Y	Y
5	Y	Y	Y
6	Y	Y	Y
7	Y	Y	Y
8	Y	Y	Y
9	Y	Y	Y
10	Y	Y	Y
11	Y	N	Y
12	Y	N	Y
13	Y	N	Y
14	Y	N	Y
15	Y	N	Y
16	Y	N	Y
17	Y	N	Y
18	Y	N	Y
19	Y	N	Y
20	Y	N	Y

2.3 Python Code

***While the code is handwritten without generative AI, it is important to note that the token generation code was written while following what was posted on Brightspace by team 6. The QoS Bandwidth portion of the code took inspiration from the python example code provided by Shabodi.

```python
import requests
import datetime

client_id = "220d0476-ecce-4a3a-a17f-e359d548d230"
client_secret = "S2JN-MDhAjS45ez1yoxuH54HKs9HNbM9DhFf1aG6xH4"
aep_host = "192.168.3.18"
access_token = None
max_bitrate_low = 100
max_bitrate_high = 5000
device_id = 7
latitude = 45.41928
longitude = -75.67868
date_time = None

#token generation
def get_token(client_id, client_secret):
    url = f"http://{aep_host}:31002/security/v1/token"

    headers = {
        "Content-Type" : "application/json"

    }
    data = {
        "client_id" : client_id,
        "client_secret" : client_secret
    }

    response_data = requests.post(url,headers=headers,json=data, verify=False)
```

Figure 3. Python Code part 1

```python
    if response_data.status_code == 200:

        token = response_data.json().get("access_token")
        #print("Success:", token)
        return token

    else:

        print("Failed", response_data.status_code, response_data.json())
        return None

#QoS reduction - setting the bandwidth
def set_bandwidth(max_bitrate):

    url = f"http://{aep_host}:7999/qos/v1/bandwidth"

    headers= {
        "Content-type": "application/json",
        "Authorization": f"Bearer {access_token}"

    }
    data = {
        "device": {
            "deviceId" : device_id
        },
        "maxBitRate": max_bitrate,
        "direction": "downlink"
    }
    response_data = requests.post(url, headers = headers, json=data, verify = False)
```

Figure 4. Python Code part 2

```
59
60
61      if response_data.status_code == 201:
62
63          print("Device " + str(device_id) + "'s Bandwidth has succesfully been changed to " + str(max_bitrate) + " Kbps \n")
64
65      else:
66
67          print("Failed to reduce Bandwidth, contact Management ASAP")
68
69
70  #generate token before API calls
71  access_token = get_token(client_id,client_secret)
72
73  #ask user for SIM swap start
74  inp = input("Enter SWAP to simulate a SIM Swap ")
75  print("\n")
76
77  if inp =="SWAP":
78      date_time = datetime.datetime.now()
79      print("SIM Swap detected in Device " + str(device_id) + " !")
80      print("Device Location: -Latitude: " + str(latitude) + "  -Longitude: " + str(longitude))
81      print("Time of SIM swap detection: " + str(date_time))
82      set_bandwidth(max_bitrate_low)
```

Figure 5. Python Code part 3

```
83
84      #user can accept or deny SIM swap
85      answer = input("Do you accept these changes y or n ? ")
86      print("\n")
87      if answer =="y":
88          set_bandwidth(max_bitrate_high)
89          print("Bitrate has been restored!")
90      else:
91          print("Bitrate will remain reduced. Please contact Management ASAP")
92
93   else:
94      print("SIM swap not initiated")
95
96
97
```

Figure 6. Python Code part 4

3 – Prototyping Test Plan (no changes)

Table 2. Prototyping Test Plan

Prototype	**Test Objective**	**Description of prototype and test method**	**Results**	**Estimated test duration**
Prototype I	Test snapshot location precision	Request the location of device via Location API multiple times and compare results for precision	Quantitative: Longitude + Latitude coordinates of device	45 minutes
Prototype II	Test snapshot info accuracy	Request device info via API multiple times and compare results accuracy	Quantitative: Snapshot info (time, UE ID) can be compared to actual values	45 minutes
Prototype III	QoS reduction efficiency testing	Trigger a QoS reduction and determine if the bitrate matches the required reduction	Quantitative: Compare bitrate to the bitrate that we are supposed to be receiving after QoS	30 minutes
Prototype IV (if time permits)	UI / UX user feedback testing	Design a mockup of different interfaces and ask people (students, TA, prof?) what they think (maybe present to client next meeting if finished)	Qualitative: User feedback from various people will help us determine what color schemes are more popular with users, button shapes, etc...	1-2 hour

4 – BOM (no changes)

Table 3. Bill of Materials

Item	Purpose	Cost per Unit ($)	Quantity Subtotal ($)
Shabodi NetAware Sandbox	Needed to learn and utilize Shabodi's APIs	0$	0$
Python 3.12.7	Needed to code the subsystems of our product	0$	0$
Microsoft VSCode	Needed to code to code in python outside of the NetAware Sandbox	0$	0$
Tkinter GUI library	GUI library for python that is easy to use and accesss and has a lot of documentation and tutorials	0$	0$

5 – GUI – Future Work

With our third prototype completed, we are now looking at creating a simple GUI that could allow users to interactively test our application. Hopefully this task will be complete, and the interface will be up and running by design day.

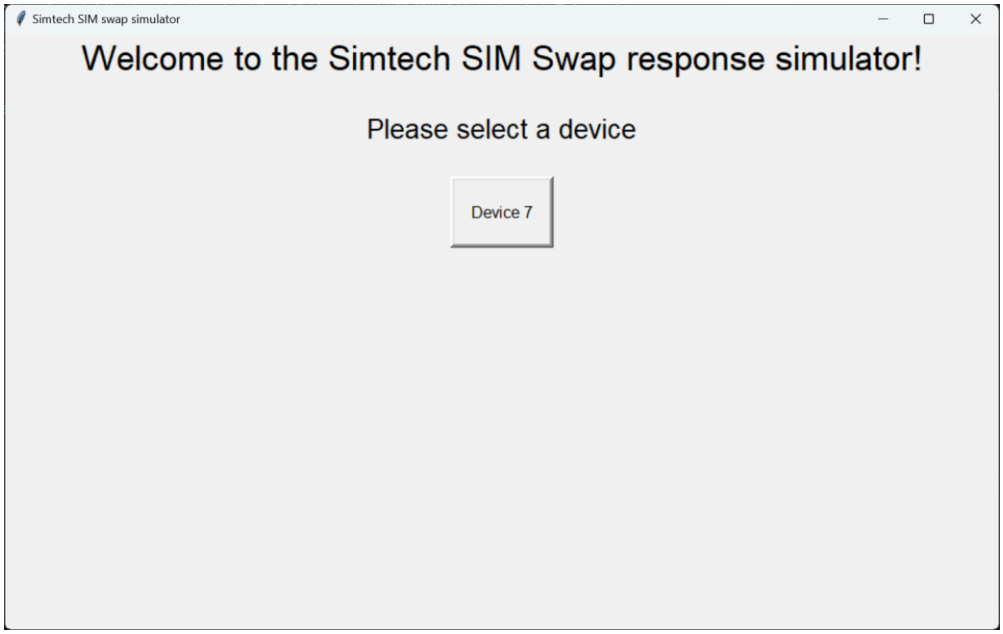

Figure 7. Initial GUI Work

6 – Conclusion

To conclude, we were successful in achieving a reduction in QoS Bandwidth using the Shabodi API. The administrator is able to accept or deny the SIM Swap and our testing shows that the application successfully responds accordingly. We will be attempting to create a simple GUI for design day.

7 – Trello Board Link

https://trello.com/b/u6dVFniw/gng1103-project