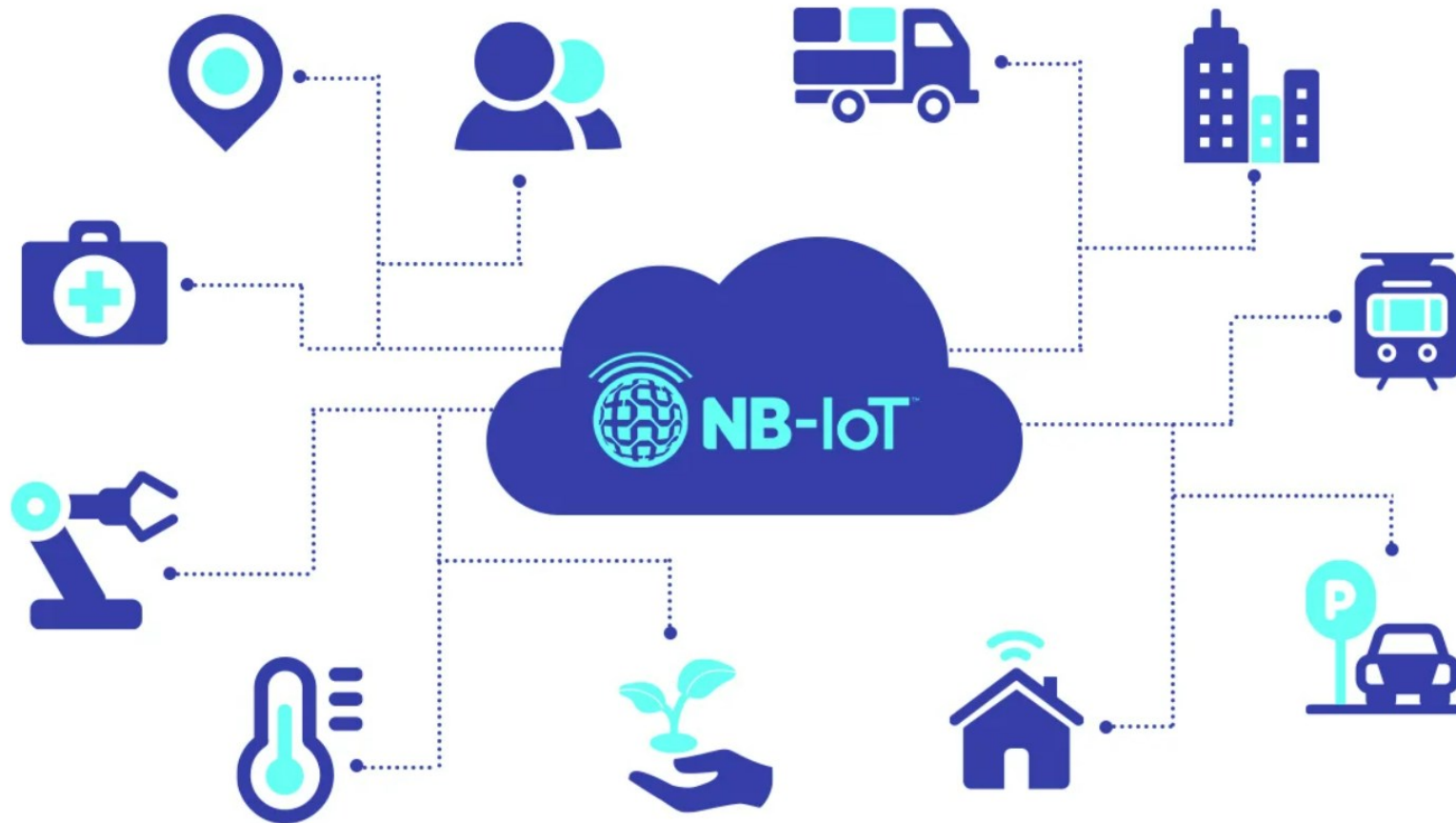


NB-IoT



NB-IoT (Narrowband Internet of Things)

- LPWAN tehnologija: Low Power Wide Area Network standard razvijen za IoT uređaje.
- Infrastruktura: Koristi licencirani opseg mobilnih operatera i SIM kartice.
- Pokrivenost: Izuzetan domet i probojnost signala (unutar zgrada i podzemno).
- Niska potrošnja: Podržava napredne modove uštede energije (PSM i eDRX).
- Hardver: Quectel BG95-M2 (NB-IoT / LTE-M mobilni modul ultra-niske potrošnje).
- Dokumentacija: Quectel BC68 set AT komandi
[http://www.elektronika.ftn.uns.ac.rs/umrezeni-embeded-sistemi/wp-content/uploads/sites/176/2018/03/Quectel BC95-GBC68 AT Commands Manual V1.5.pdf](http://www.elektronika.ftn.uns.ac.rs/umrezeni-embeded-sistemi/wp-content/uploads/sites/176/2018/03/Quectel_BC95-GBC68_AT_Commands_Manual_V1.5.pdf)

NB-IoT ESP32 aplikacija

- <https://github.com/AnjaTanovic/Umrezeni-embedded-sistemi/tree/main/Vezba-10>
- **Inicijalizacija sistema:** Pokretanje ESP32 i NB-IoT modema (*BG600L_turnON()*).
- **Mrežna registracija:** Povezivanje modula na NB-IoT mrežu operatera (*BG600L_nwkSetup()*).
- **Slanje podataka:** Uspostavljanje UDP komunikacije sa ciljanim serverom i transmisija paketa (payload) ka serveru (*BG600L_TxUDPbin()*).
- **Prijem odgovora:** Prihvatanje i provera povratnog paketa sa servera (Echo test / *BG600L_RxUDPbin()*).
- **Ušteda energije (Modem):** Slanje modema u Power Saving Mode uz zadržavanje mrežne registracije (*BG600L_enterPSM()*).
- **Ušteda energije (MCU):** Prelazak ESP32 mikrokontrolera u Deep Sleep režim na 5 minuta.
- **Ciklično izvršavanje:** Automatsko buđenje sistema i ponavljanje celog algoritma.

Serverska aplikacija - Echo server (UDP)

- Python skripta `udp_echo.py` konstantno osluškuje dolazni saobraćaj na udaljenom serveru preko UDP protokola.
- Server prima paket u binarnom formatu, parsira tekst, prebacuje sve karaktere u velika slova (*kapitalizovana verzija*) i vraća ih nazad uređaju kao *Echo test* (potvrda dvosmerne komunikacije).
- Port na serveru je fiksno konfigurisan u kodu (usklađen sa portom koji poziva ESP32 mikrokontroler).
- Pokretanje servera:

`$ python3 udp_echo.py`

```
from socket import *
import time

port = 50123
serverSocket = socket(AF_INET, SOCK_DGRAM)
serverSocket.bind(('', port))
print ('Echo server is ready to receive (port ' + str(port) + ')\n')

msgCnt = 1
while True:
    try:
        messageIn, clientAddress = serverSocket.recvfrom(4096)

        ts = time.localtime()
        print('\033[0;34;40mEcho server (' + str(port) + ') Msg#', str(msgCnt))
        print(time.strftime('%Y-%m-%d %H:%M:%S\033[0;37;40m', ts))
        print('    Rx: ', messageIn)
        print('    Tx: ', messageIn.upper())
        serverSocket.sendto(messageIn.upper(), clientAddress)

        msgCnt += 1
    except:
        print('ERROR in Echo UDP')
```

Echo server (TCP)

```
import socket

HOST = ''                # Symbolic name meaning all available interfaces
PORT = 50123             # Arbitrary non-privileged port
NUM_OF_CLIENTS = 1

tcpSocket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
tcpSocket.bind((HOST, PORT))
tcpSocket.listen(NUM_OF_CLIENTS)

print ('Echo server is ready to receive (port ' + str(PORT) + ')\n')

msgCnt = 1
while True:
    try:
        conn, addr = tcpSocket.accept()
        print ('Connected by', addr)
        while True:
            data_in = conn.recv(1024)
            if not data_in:
                break
            data_out = data_in.upper()
            print ('    Rx: ', data_in)
            print ('    Tx: ', data_out)
            conn.sendall(data_out)
    except:
        conn.close()
        print('ERROR in Echo TCP')
```

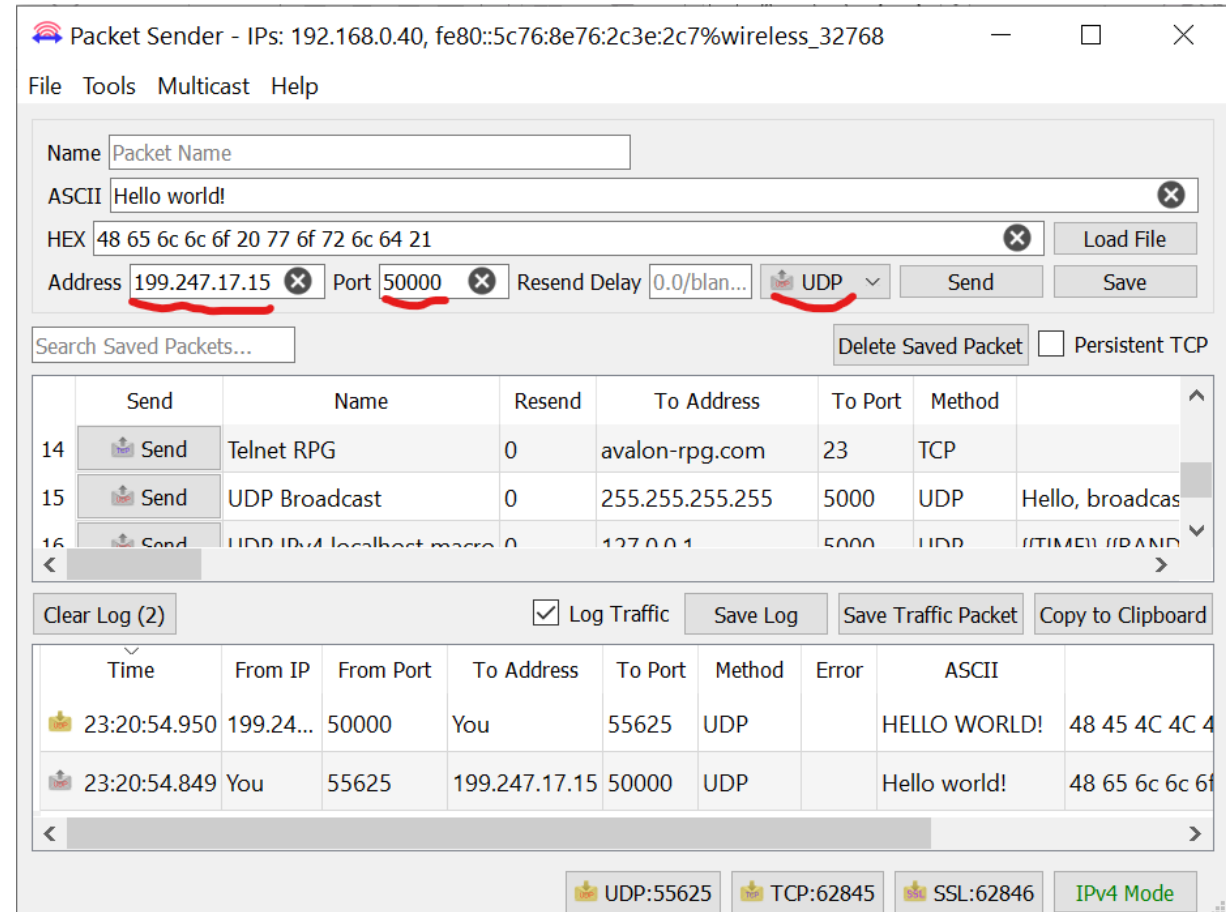
Packet Sender – pomoćni alat za slanje paketa

- https://github.com/dannagle/PacketSender/releases/download/v7.2.3/PacketSenderPortable_v7.2.3.zip



```
UES student.tlp - test.e01@199.247.17.15:...
test.e01@UES_server:~$ python3 udp_echo.py
Echo server is ready to receive (port 50000)

Echo server (50000) Msg# 1
2021-06-07 23:20:54
  Rx: b'Hello world!'
  Tx: b'HELLO WORLD!'
```



Packet Sender - IPs: 192.168.0.40, fe80::5c76:8e76:2c3e:2c7%wireless_32768

File Tools Multicast Help

Name: Packet Name

ASCII: Hello world!

HEX: 48 65 6c 6c 6f 20 77 6f 72 6c 64 21

Address: 199.247.17.15 Port: 50000 Resend Delay: 0.0/blank UDP Send Save

Search Saved Packets... Delete Saved Packet Persistent TCP

	Send	Name	Resend	To Address	To Port	Method	
14		Telnet RPG	0	avalon-rpg.com	23	TCP	
15		UDP Broadcast	0	255.255.255.255	5000	UDP	Hello, broadcas
16		UDP IPv4 localhost mcast	0	127.0.0.1	5000	UDP	((TIMES)) ((RAND

Clear Log (2) Log Traffic Save Log Save Traffic Packet Copy to Clipboard

	Time	From IP	From Port	To Address	To Port	Method	Error	ASCII	
	23:20:54.950	199.24...	50000	You	55625	UDP		HELLO WORLD!	48 45 4C 4C 4
	23:20:54.849	You	55625	199.247.17.15	50000	UDP		Hello world!	48 65 6c 6c 6f

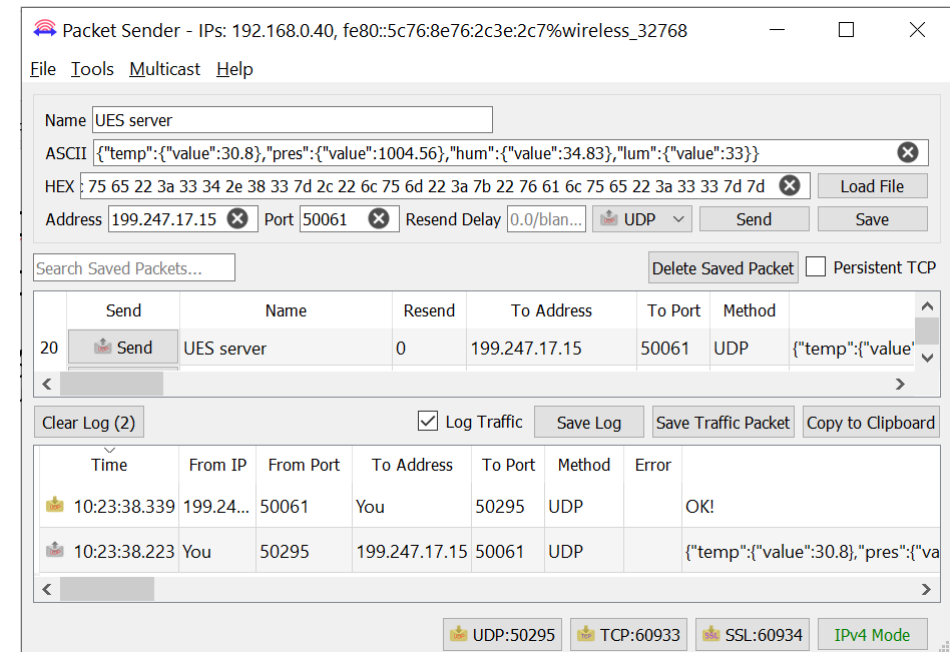
UDP:55625 TCP:62845 SSL:62846 IPv4 Mode

Primer: Upis sadržaja UDP paketa u bazu

- U ovom primeru, sadržaj UDP paketa (očitavanja senzora u JSON formatu) upisuje se u bazu podataka, u postojeću tabelu **merenja**

PRIMER:

```
{
  "temp": {
    "value": 30.8
  },
  "pres": {
    "value": 1004.56
  },
  "hum": {
    "value": 34.83
  },
  "lum": {
    "value": 33
  }
}
```



```
test.e01@UES_server:~$ python3 udp_senzor.py
UDP server (port 50061)

UDP server (50061) Poruka# 1
2021-06-09 10:14:33
  Temperatura:      30.8 °C
  Pritisak:         1004.56 mBar
  Vlaznost vazduha: 34.83 %
  Osvetljenost:     33 lux
```



Showing rows 0 - 3 (4 total, Query took 0.0003 seconds.)

SELECT * FROM `merenja`

Profiling [Edit inline] [Edit] [Explain SQL]

Show all | Number of rows: 25 | Filter rows: Search this table | Sort by key: None

	br_merenja	vreme	tip_senzora	vrednost
☐ Edit Copy Delete	1	2021-06-09 08:14:33	temperatura	30.8
☐ Edit Copy Delete	2	2021-06-09 08:14:33	pritisak	1004.56
☐ Edit Copy Delete	3	2021-06-09 08:14:33	vlaznost	34.83
☐ Edit Copy Delete	4	2021-06-09 08:14:33	osvetljenost	33

```

from socket import *
import time
import mysql.connector
import json

mydb = mysql.connector.connect(
    host = "localhost",
    user = "test.e01",
    password = "...",
    database = "db_test_e01"
)

port = 50061
serverSocket = socket(AF_INET, SOCK_DGRAM)
serverSocket.bind(('', port))
print ('UDP server (port ' + str(port) + ')\n')

msgCnt = 1
while True:
    try:
        messageIn, clientAddress = serverSocket.recvfrom(4096)

        ts = time.localtime()
        print('\033[0;34;40mUDP server (' + str(port) + ') Poruka#', str(msgCnt))
        print(time.strftime('%Y-%m-%d %H:%M:%S\033[0;37;40m', ts))

        data = json.loads(messageIn.decode("utf-8"))
        print ("    Temperatura:      ", data["temp"]["value"], "\u00b0C")
        print ("    Pritisak:           ", data["pres"]["value"], "mBar")
        print ("    Vlaznost vazduha:    ", data["hum"]["value"], "%")
        print ("    Osvetljenost:       ", data["lum"]["value"], "lux")

        mycursor = mydb.cursor()
        mycursor.execute("INSERT INTO merenja (tip_senzora, vrednost) VALUES ('temperatura'," + str(data["temp"]["value"]) + ")")
        mycursor.execute("INSERT INTO merenja (tip_senzora, vrednost) VALUES ('pritisak'," + str(data["pres"]["value"]) + ")")
        mycursor.execute("INSERT INTO merenja (tip_senzora, vrednost) VALUES ('vlaznost'," + str(data["hum"]["value"]) + ")")
        mycursor.execute("INSERT INTO merenja (tip_senzora, vrednost) VALUES ('osvetljenost'," + str(data["lum"]["value"]) + ")")
        mydb.commit()

        serverSocket.sendto(bytearray("OK!", "utf-8"), clientAddress)

        msgCnt += 1
    except:
        print('Greska!!!')

```

Zadatak

- Modifikovati ESP32 aplikaciju tako da na pritisak tastera šalje trenutna očitavanja temperature i vlažnosti vazduha sa Temp&Hum senzora. Takođe, modifikovati serversku skriptu kako bi primala novu strukturu podataka i uspešno skladištila očitavanja u bazu.
- Kreirati web stranicu koja grafički prikazuje pristigle rezultate očitavanja senzora.