\*\*\*\*decode/nodelogparser\_serial.py

Décode le payload depuis l’uart du nœud lora

Configuration du json pour envoyer les logs et la trame Lora dans le port serie.



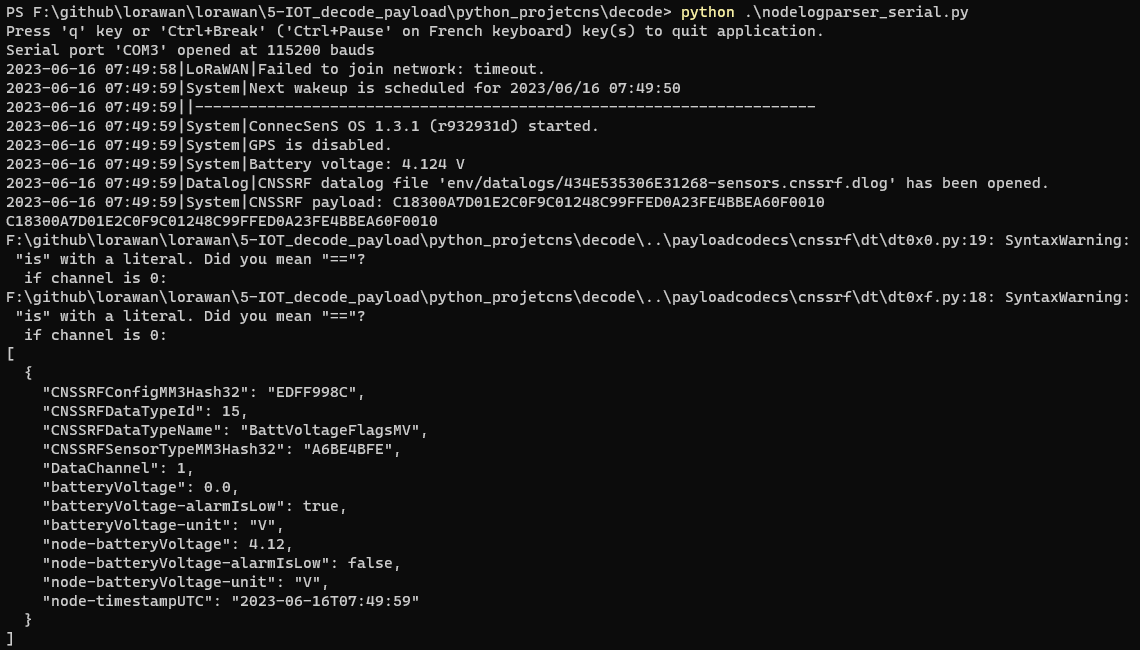
Paramètre du com3 : 115 200 bauds

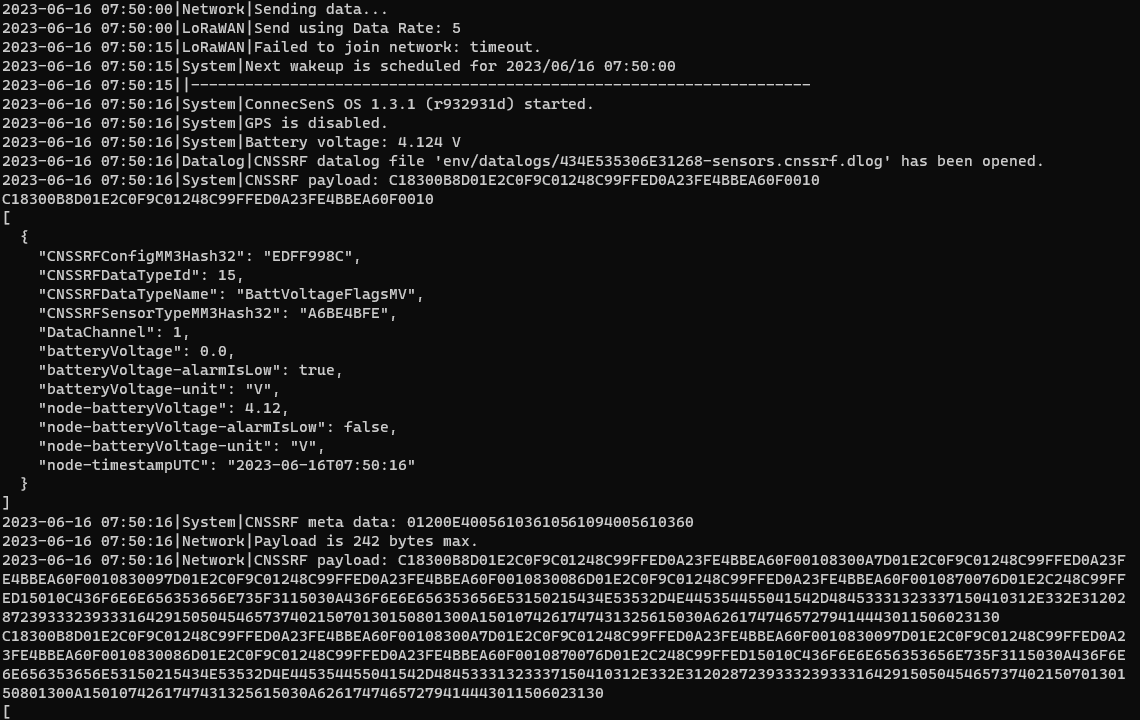
8 bits

Pas de parité

1 bit de stop

Pas de contrôle de flux matériel

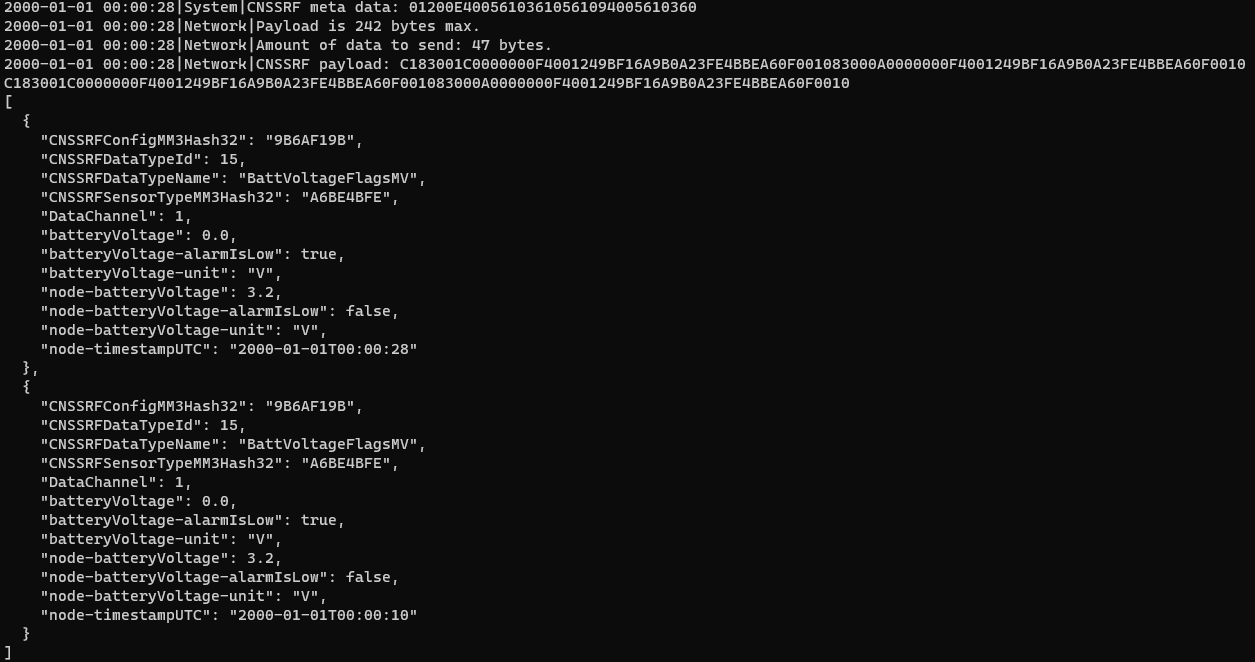




En dernier nous avons un payload plus important qui correspond à 14 trame json

Décodage de la valeur du payload :

1 payload :



2 payload

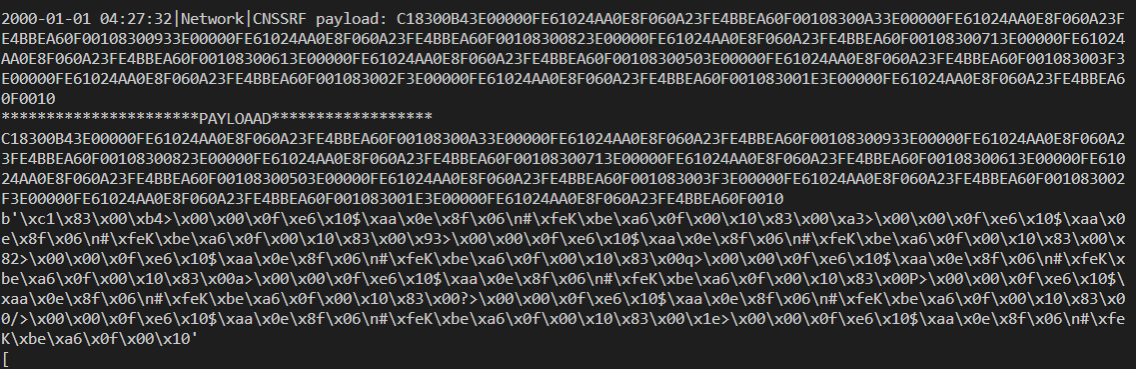
Une donnée trame json correspond environ à 24bytes (octets) la limite est de 242 octets maximums environ 10 trames json.

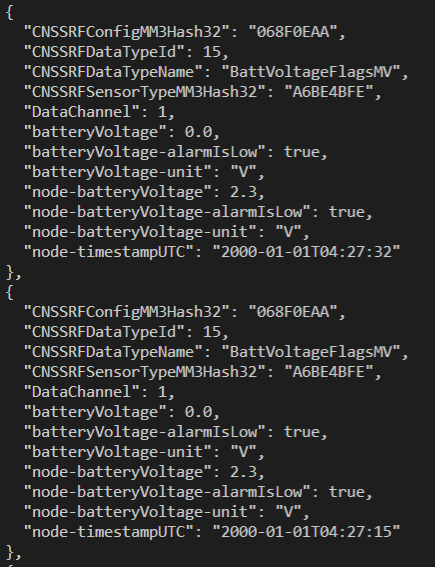


Ici nous avons 14 données json à 23.5\*14=329 mais limite le payload à 242byte c’est pour ça que nous avons 2 payload. Nous avons aussi des CNSSRFDataTypeName : config aussi

Programme Python

Decode le Payload





Se programme fait appelle au payloadcodecs celui utilisé pour décoder le payload du système.

Le script nodelogparser\_serial.py réceptionne les données du nœud via l’uart.

**La fonction process\_log\_line(line)**

Execution:

**payload = bytes.fromhex(payload)**

1. On transforme les bytes en string et on enlève le \r\n. # line = bytes\_to\_string(line).rstrip('\r\n')
2. On positionne le curseur de lecture au niveau du payload +1 pour l’espace.
3. On transforme notre string payload (bytes.fromhex(payload))

pour créer un objets bytes à partir d’une chaîne de chiffres hexadécimaux.

"01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e 0f"

b'\x01\x02\x03\x04\x05\x06\x07\x08\t\n\x0b\x0c\r\x0e\x0f'

Explication ++

hexstring = "bb 0D 02 55 55 5A ce"

b'\xbb\x0d\x02\x55\x55\x5a\xce' == b'\xbb\r\x02UUZ\xce'

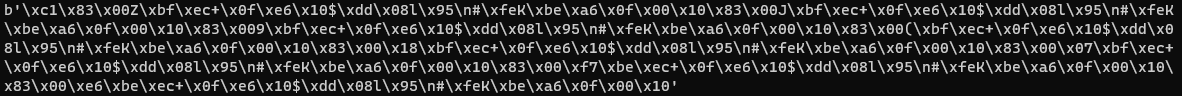
>>b'\xbb\r\x02UUZ\xce'.hex()

'bb0d0255555ace'

Python affiche automatiquement les octets qui peuvent être interprétés comme ASCII imprimables comme les caractères qu’ils impriment chr(0x55) == U c’est la même donnée



**res = cnssrf\_frame.decode\_payload(payload**):



res = cnssrf\_frame.decode\_payload(payload)

print(json.dumps(res, sort\_keys = True, indent = 2, ensure\_ascii = to\_ascii))

\*\*\*\*decode/file\_payload/rfpayloaddecoder.py

\*\*\*\*decode/mqtt/.py