**Node RED Program**

Installation Guide

Release: 1.0.3

February 2020

This document describes how to install and configure the Node RED package version **1.0.3** using NodeJS **12.2.0** that comes with npm version **6.9.0.** It includes the following sections:

* Installation Resources
* Preinstallation considerations
* Installation Process

Installation Resources

Zack’s Node-RED Package:

<https://github.com/imZack/pkg-node-red>

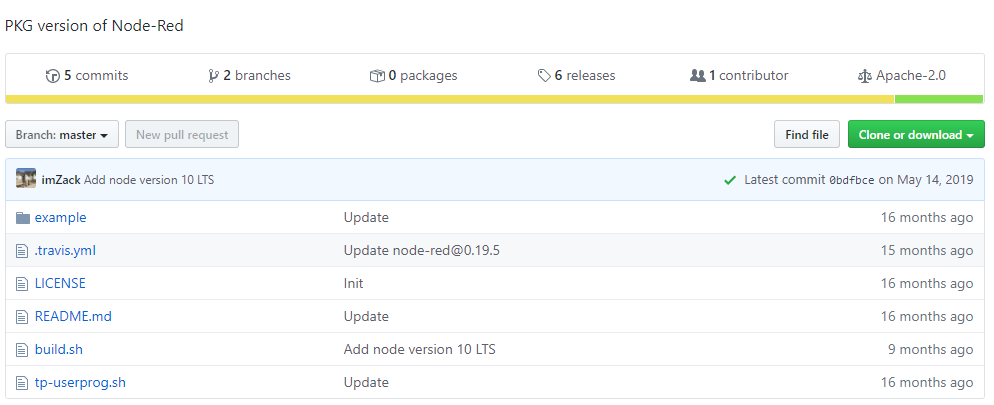
Travis CI of Zack’s package:

<https://www.travis-ci.com/imZack/pkg-node-red/builds/111784956#L3757>

Package fetch used automatically in our process for uploaded-v2.6-node-v12.2.0-linux-armv7 (Notice that the 10.17.0 Nodejs version is not available):

<https://github.com/zeit/pkg-fetch/releases>

Preinstallation considerations

In order to install the versions listed before of Nodejs and Node-RED, some changes are needed to be done through Zack’s package as follow:

**.travis.yml**

sudo: false

language: node\_js

node\_js:

- '**12**'

install:

- npm install -g pkg

script:

- bash build.sh

- bash tp-userprog.sh

- ls -alh ./release

env:

- NODERED\_VERSION=**1.0.3**

deploy:

provider: releases

api\_key:

secure: 

file\_glob: true

file: release/\*

skip\_cleanup: true

on:

repo: imZack/pkg-node-red

tags: true

**Build.sh**

#!/bin/bash

set -x

npm install "node-red@$NODERED\_VERSION"

node -e "const data = require('./node\_modules/node-red/package.json'); data.pkg = { assets: ['./\*\*/\*'] }; require('fs').writeFileSync('new-package.json', JSON.stringify(data, null, ' '));"

cp new-package.json node\_modules/node-red/package.json

(

cd node\_modules/node-red || exit 1

npm install --production

pkg \

--targets **node12.2.0-linux-armv7,node12.2.0-linux-x64,node12.2.0-win-x64,node12.2.0-macos-x64 \**

--out-path ../../release --public .

)

**tp-userprog.sh**

**no changes so far in this bash script**

#!/bin/bash

set -x

## Create armv7

(

mkdir userprog-armv7

cp -r example/\* userprog-armv7

cp release/node-red-linux-armv7 userprog-armv7/node-red

chmod +x userprog-armv7/node-red

cd userprog-armv7 || exit 1;

tar czvf ../release/tp-node-red-armv7.tar.gz .

)

## Create x64

(

mkdir userprog-x64

cp -r example/\* userprog-x64

cp release/node-red-linux-x64 userprog-x64/node-red

chmod +x userprog-x64/node-red

cd userprog-x64 || exit 1;

tar czvf ../release/tp-node-red-x64.tar.gz .

)

Once all the changes made in this package, we can create all the files on our gateway to /Home/Moxa.

Installation Process

1. **Install NodeJS 12.2.0, npm 6.9.0 and nvm 0.35.2**

Make sure you are connected in order to update the gateway packages before any install and start running the following commands:

moxa@Moxa:~$ sudo apt-get update

moxa@Moxa:~$ sudo apt-get install curl software-properties-common

MUST BE ROOT TO EXECUTE THIS FOLLOWING COMMAND ONLY

moxa@Moxa:~$ sudo su

root@Moxa:/home/moxa# sudo npm install -g n

root@Moxa:/home/moxa# $curl -L https://raw.githubusercontent.com/tj/n/master/bin/n -o n

bash n 12.2.0

GO BACK TO /HOME/MOXA (Ctrl-D) AND RUN GCC INSTALL AS RECOMMENDED RIGHT AFTER NODEJS INSTALLATION

moxa@Moxa:~$ sudo apt-get install gcc g++ make

NVM INSTALL

moxa@Moxa:~$ curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.35.2/install.sh | bash

moxa@Moxa:~$ nvm –version

1. **Create Zack’s node RED package files**

moxa@Moxa:~$ nano .travis.yml (copy and paste changes highlighted in the previous section)

moxa@Moxa:~$ nano build.sh (copy and paste changes highlighted in the previous section)

moxa@Moxa:~$ nano tp-userprog.sh (copy and paste code)

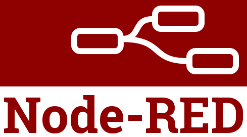
1. **Install a Package using npm**

moxa@Moxa:~$ sudo npm install -g pkg

npm WARN deprecated request@2.88.2: request has been deprecated, see https://github.com/request/request/issues/3142

/usr/local/bin/pkg -> /usr/local/lib/node\_modules/pkg/lib-es5/bin.js

+ pkg@4.4.3

added 123 packages from 152 contributors in 100.115s

1. **Install Node-RED**

In this step we are installing node-RED by executing build.sh first to build our package that will include Node-RED and Nodejs and then we can proceed to create a tar.gz file of our package to upload on Thingspro.

moxa@Moxa:~$ export NODERED\_VERSION=1.0.3

moxa@Moxa:~$ bash build.sh (this command should take 20-30 minutes)

moxa@Moxa:~$ bash tp-userprog.sh

**BECAUSE SOME OF ZACK’s CODE FAILED IN OUR CASE CREATING THE CERTS AND DATA FOLDERS IN BUILD.SH SCRIPT, AND END UP CREATING A TAR.GZ FILE IN TP-USERPROG.SH SCRIPT WITHOUT CERTAIN REQUIRED TYPES OF FOLDERS AND FILES, WE ARE GOING TO CREATE OUR PROGRAM TAR.GZ FILE MANUALLY USING THE FOLLOWING STEP:**

1. **Create a tar.gz file of our program**

Use some of Zack’s Node-RED package resources in folder /example to download or copy paste the following folders and files in /Home/Moxa/userprog-armv7:

* **/certs**
  + cert.pem
  + key.pem
* **/data**,
* **/lib/flows/**EMPTY
* **package.json** originally named new-package.json that was created through our installation process in /home/moxa.
* **package-lock.json** that was also created through our installation process in /home/moxa.
* **/node\_modules.**

And the two files:

* **exec**,
* **settings.js.**

Now our program is ready to be compiled using the following command:

moxa@Moxa:~$ tar -cvf - exec node-red settings.js data/ certs/ | gzip > nodered-pro-settings.tar.gz