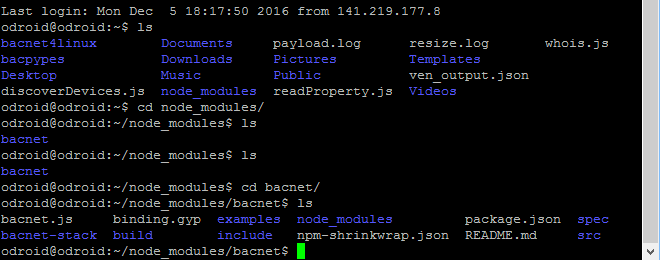
**Bacnet Javascript (Node.js) Information**

1. It is basically a bacnet stack written in ‘C’ and wrapped for Node.js
2. Can be downloaded from: <https://www.npmjs.com/package/bacnet>
3. Checked for working properly on Linux System
4. Installation Procedure:
   1. Open Terminal
   2. Type and run: *npm install bacnet*
5. Directory structure of installed stack is as follows:



When installed on Odroid, the installed stack is shown as per above image.

## Contents

* bacnet-stack/ - contains the full contents of the c library checked out from svn://svn.code.sf.net/p/bacnet/code/trunk/bacnet-stack
  + .svn/ - the svn db is checked in to make it easier to update to newer versions and in case we have something to push back to the c library - I don't know if this is a bad idea
* src/ - contains the wrapper code
  + h\_\*.c - request and response message handlers - modified from code in bacnet-stack/demo/handlers
  + functions.cc - contains the addon code that runs when functions are called from js
  + init.cc - runs on module initialisation
  + module.cc - defines the module
  + \*conversion.cc - provides conversions between js and the BACnet structs
  + emitter.cc - emits js events into an EventEmitter in js-land asynchronously with events from the c code
* examples/ - contains example script which uses the module

Implemented Functions:

* whois
* write property
* read property

Implemented handle to receive these messages:

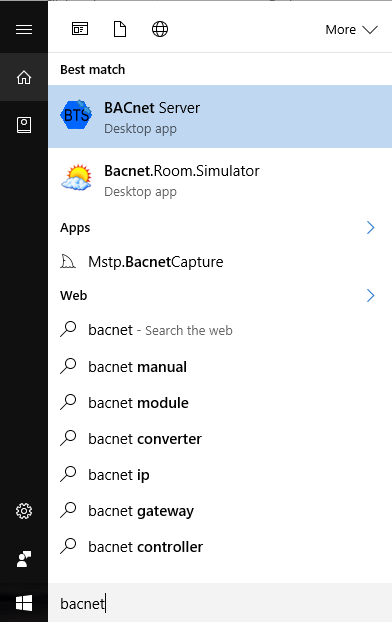
* i am
* read ack
* write ack

For detailed information, visit: <https://www.npmjs.com/package/bacnet>

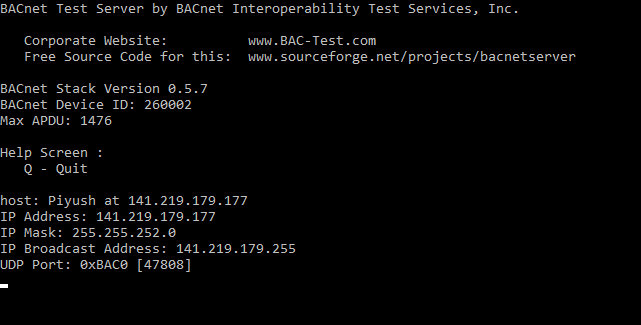
**How to run examples**

Server must be downloaded and installed from link as: <https://sourceforge.net/projects/bacnetserver/>

Run the server as shown in below image.



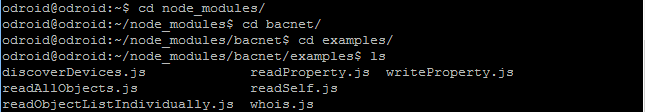
After the server is started, following screen will appear.



**Device ID of server**

**IP of server**

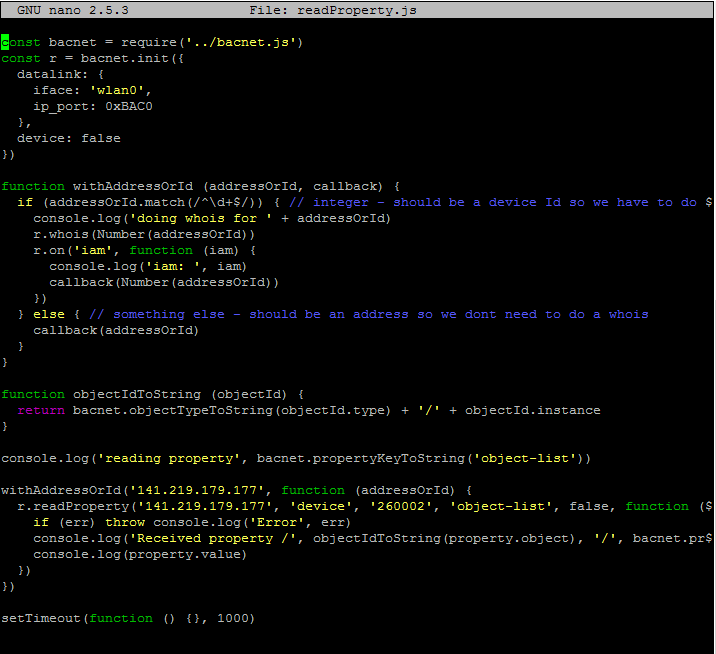
Open terminal window and enter examples directory as below:



Open and edit the example file to be run. For ex. To edit example named readProoperty.js do as follows:



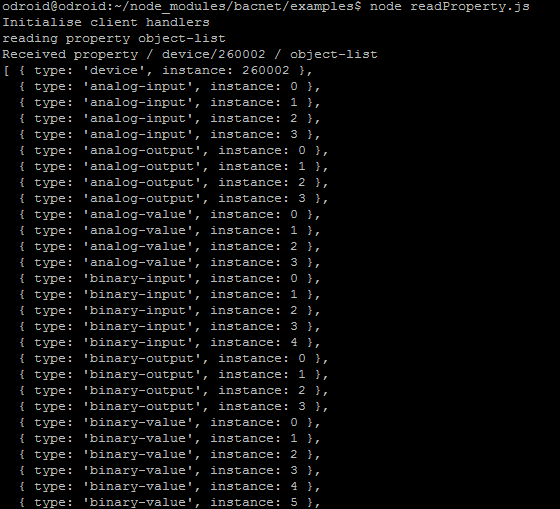
Edit the file as highlighted in figure below.

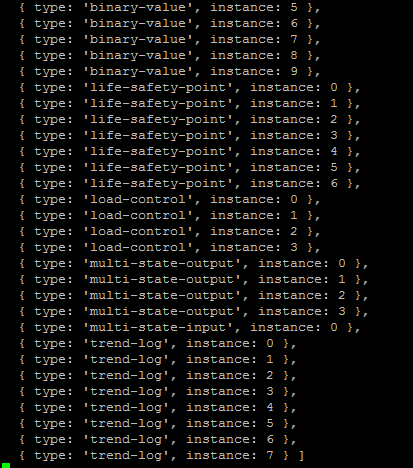


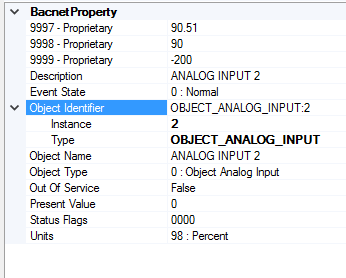
**Device ID must be of server**

**IP must be of server**

Now run the example as per below image:

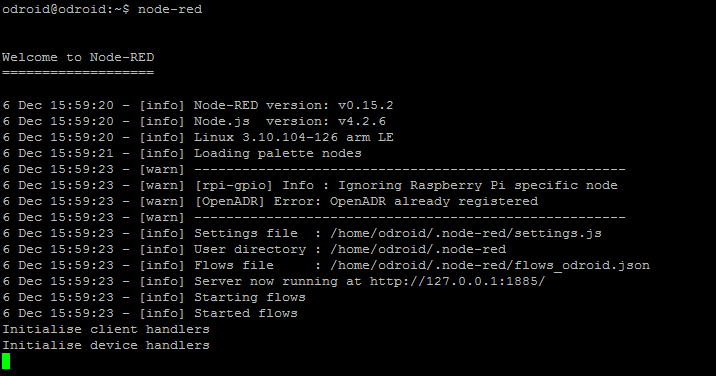






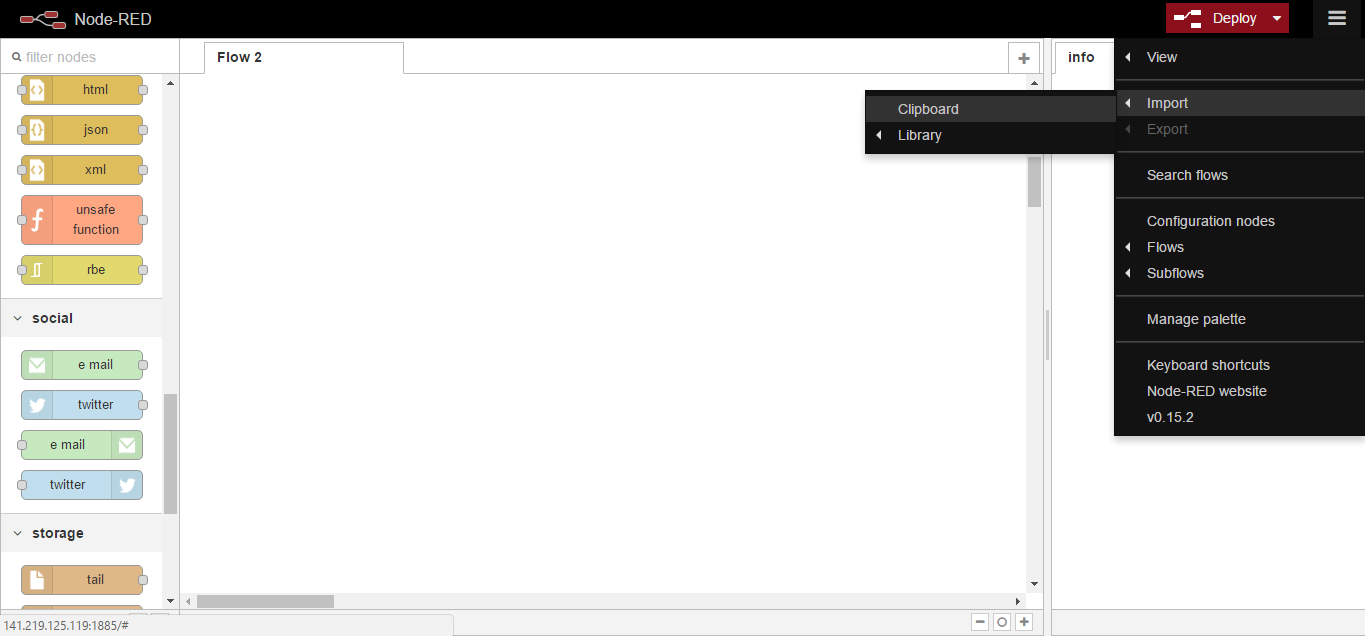
**Node-RED Implementation**

After the stack is installed, Run node-red on the device.

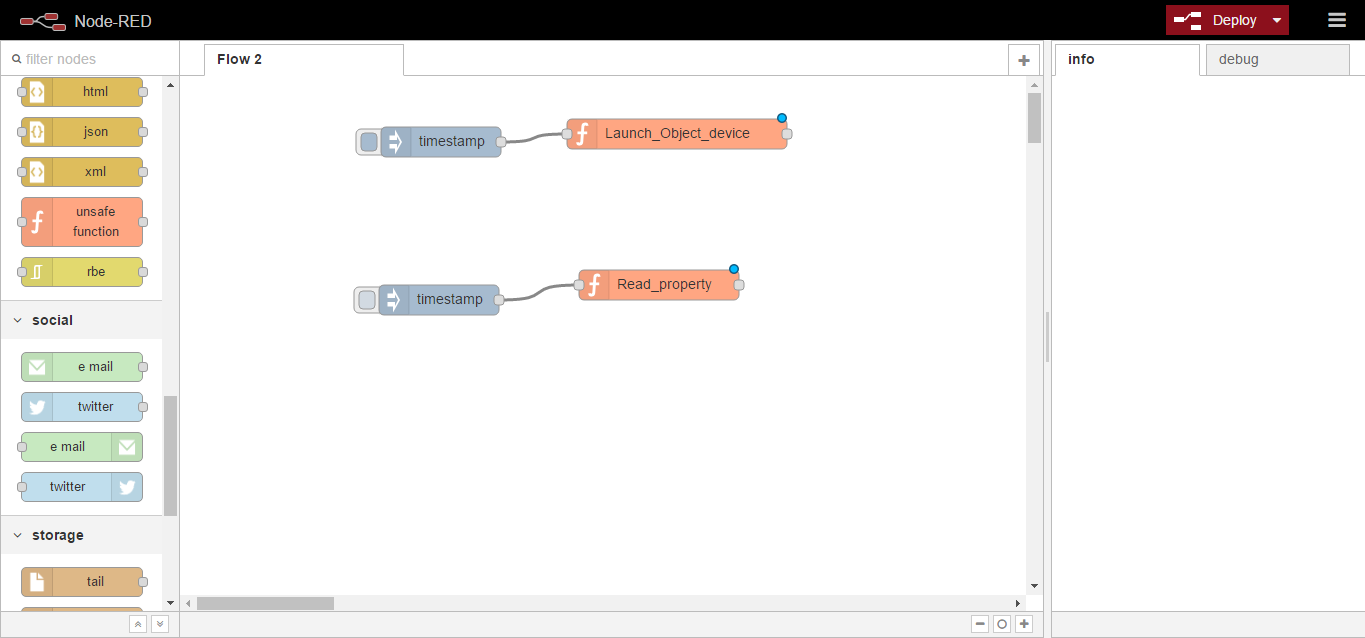


[{"id":"5446abb8.1fb4f4","type":"inject","z":"946af227.86ee5","name":"","topic":"","payload":"","payloadType":"date","repeat":"","crontab":"","once":true,"x":261,"y":66,"wires":[["f4b39d6.b88946"]]},{"id":"f4b39d6.b88946","type":"unsafe-function","z":"946af227.86ee5","name":"Launch\_Object\_device","func":"'use strict'\nconst async = require('async')\nconst bacnet = require('bacnet')\nconst r = bacnet.init({\n datalink: {\n iface: 'wlan0',\n ip\_port: 0xBAC0\n },\n device: true\n})\n\n\nr.whois('141.219.219.6',260001,260003)\nr.on('iam', function (iam) {\n console.log('iam: ', iam)\n\n})\n","outputs":1,"noerr":0,"x":497,"y":58,"wires":[[]]},{"id":"8532c69f.3820f8","type":"unsafe-function","z":"946af227.86ee5","name":"Read\_property","func":"const bacnet = require('bacnet')\nconst r = bacnet.init({\n datalink: {\n iface: 'wlan0',\n ip\_port: 0xBAC0\n },\n device: false\n})\n\nfunction withAddressOrId (addressOrId, callback) {\n if (addressOrId.match(/^\\d+$/)) { // integer - should be a device Id so we have to do a whois\n console.log('doing whois for ' + addressOrId)\n r.whois(Number(addressOrId))\n r.on('iam', function (iam) {\n console.log('iam: ', iam)\n callback(Number(addressOrId))\n })\n } else { // something else - should be an address so we dont need to do a whois\n callback(addressOrId)\n }\n}\n\nfunction objectIdToString (objectId) {\n return bacnet.objectTypeToString(objectId.type) + '/' + objectId.instance\n}\n\nconsole.log('reading property', bacnet.propertyKeyToString('object-list'))\n\nwithAddressOrId('141.219.219.6', function (addressOrId) {\n r.readProperty('141.219.219.6', 'device', '260002', 'object-list', false, function (err, property) {\n if (err) throw console.log('Error', err)\n console.log('Received property /', objectIdToString(property.object), '/', bacnet.propertyKeyToString(property.property))\n console.log(property.value)\n })\n})\n\nsetTimeout(function () {}, 1000)\n","outputs":1,"noerr":0,"x":479,"y":209,"wires":[[]]},{"id":"32c52be6.05af44","type":"inject","z":"946af227.86ee5","name":"","topic":"","payload":"","payloadType":"date","repeat":"","crontab":"","once":false,"x":259,"y":224,"wires":[["8532c69f.3820f8"]]}]

Copy the above code and paste into Node-RED import from clipboard area as in figure below:



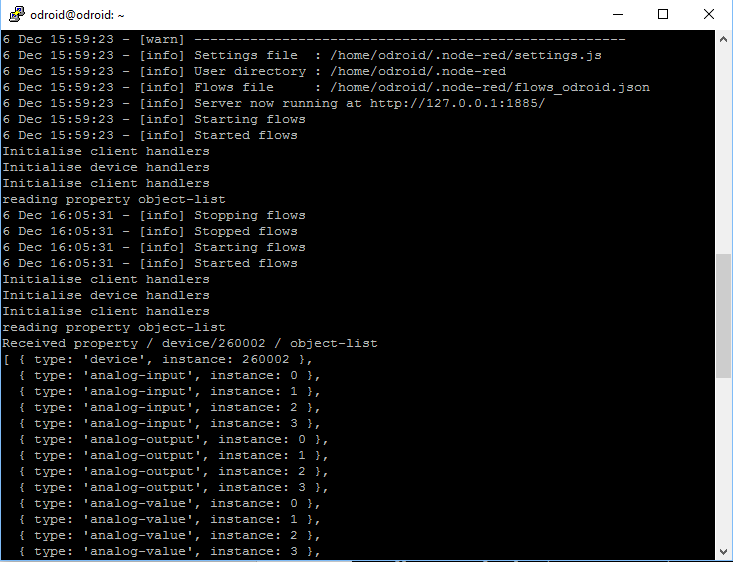
Following flow will appear:

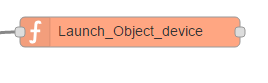


Click here

Click on inject button as shown in above figure.

Following text will appear in console window.





JavaScript written in above function node is shown below:

'use strict'

const async = require('async')

const bacnet = require('bacnet')

const r = bacnet.init({

datalink: {

iface: 'wlan0',

ip\_port: 0xBAC0

},

device: true

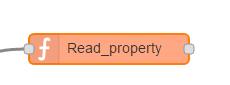
})

r.whois('141.219.219.6',260001,260003)

r.on('iam', function (iam) {

console.log('iam: ', iam)

})



JavaScript written in above function node is shown below:

const bacnet = require('bacnet')

const r = bacnet.init({

datalink: {

iface: 'wlan0',

ip\_port: 0xBAC0

},

device: false

})

function withAddressOrId (addressOrId, callback) {

if (addressOrId.match(/^\d+$/)) { // integer - should be a device Id so we have to do a whois

console.log('doing whois for ' + addressOrId)

r.whois(Number(addressOrId))

r.on('iam', function (iam) {

console.log('iam: ', iam)

callback(Number(addressOrId))

})

} else { // something else - should be an address so we dont need to do a whois

callback(addressOrId)

}

}

function objectIdToString (objectId) {

return bacnet.objectTypeToString(objectId.type) + '/' + objectId.instance

}

console.log('reading property', bacnet.propertyKeyToString('object-list'))

withAddressOrId('141.219.219.6', function (addressOrId) {

r.readProperty('141.219.219.6', 'device', '260002', 'object-list', false, function (err, property) {

if (err) throw console.log('Error', err)

console.log('Received property /', objectIdToString(property.object), '/', bacnet.propertyKeyToString(property.property))

console.log(property.value)

})

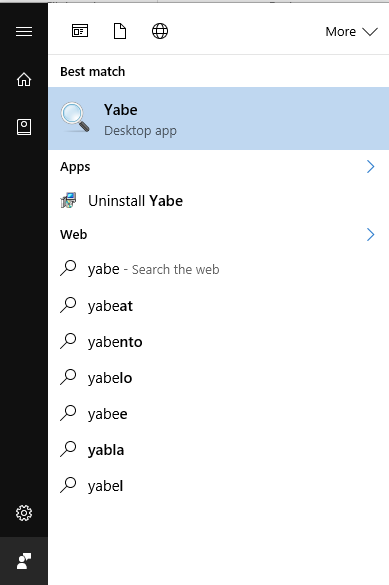
})

setTimeout(function () {}, 1000)

**Validation:**

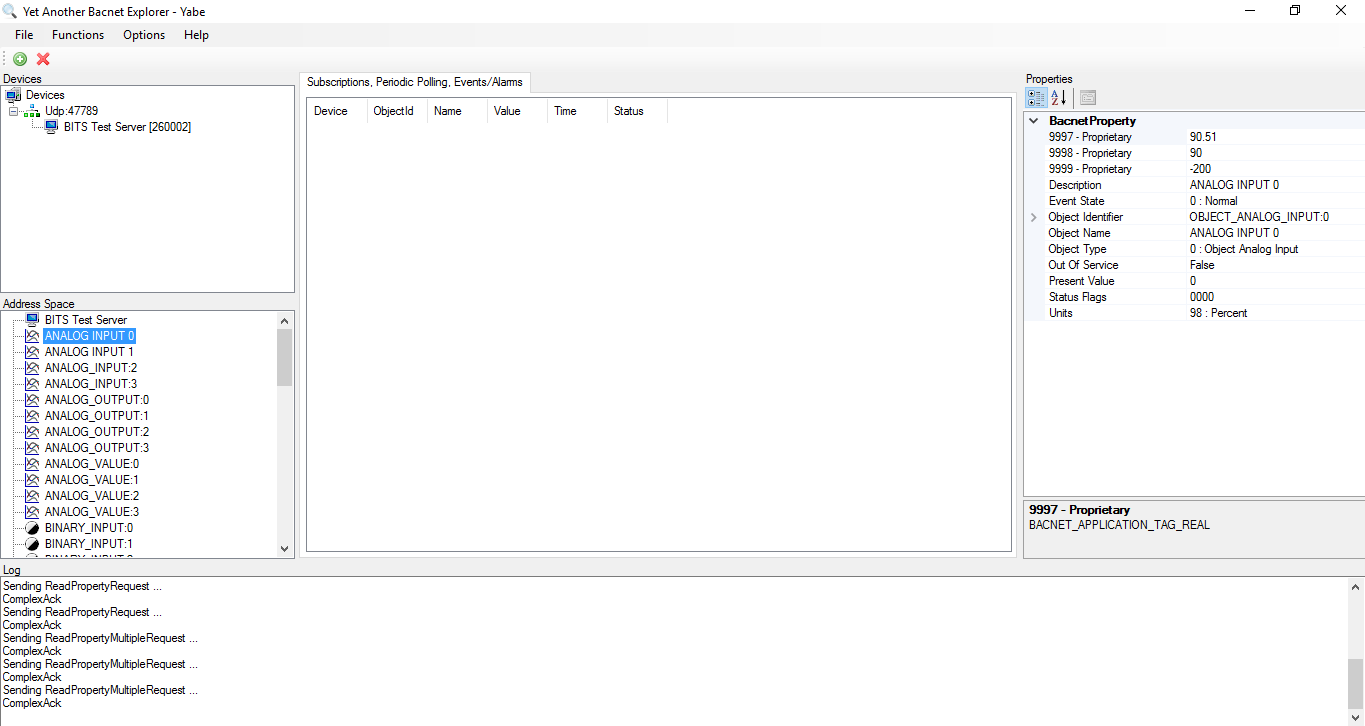
Download and install YABE. <https://sourceforge.net/projects/yetanotherbacnetexplorer/>

Run YABE:



Following screen will appear:

Compare details in below screen with the data in node-RED console window.



Properties

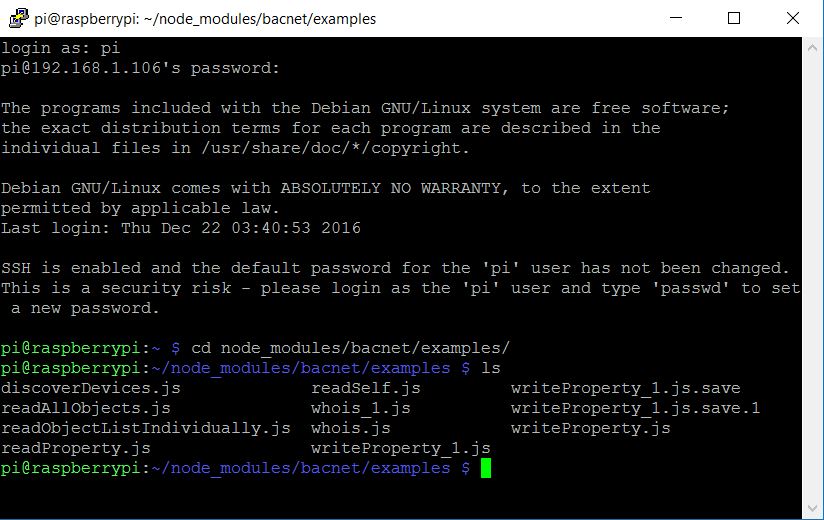
OBJECTS

**Write property service implementation**

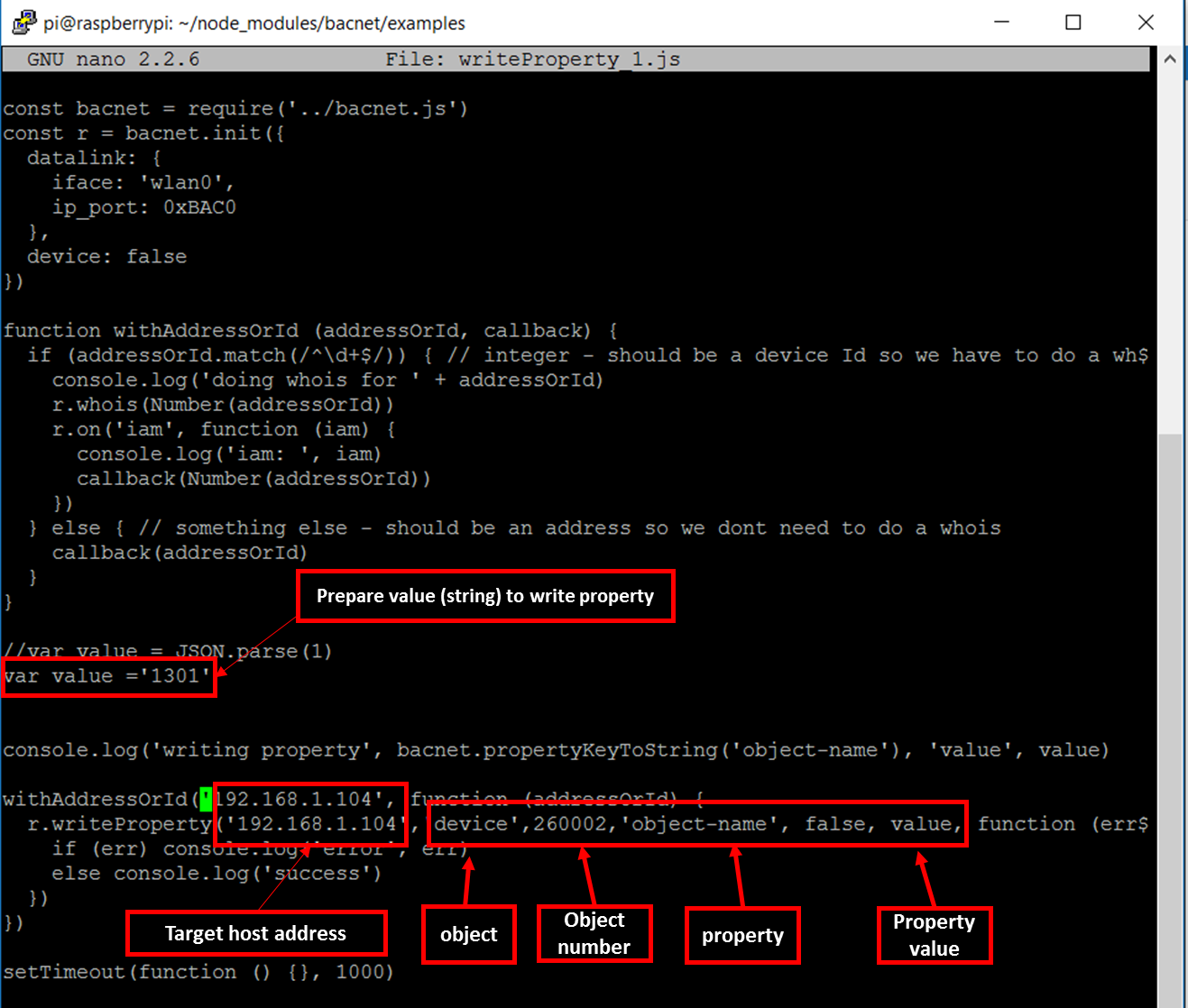
Writeproperty.js

Firstly, get access to your single board computer (odriod/Raspberry pi). Enter the bacnet library by:

$:cd node\_modules/bacnet/examples



A sample JavaScript code of Write property.js has been provided. Some modification must be made to make it work. (nano writeProperty.js)



1. var value = ‘xxx’

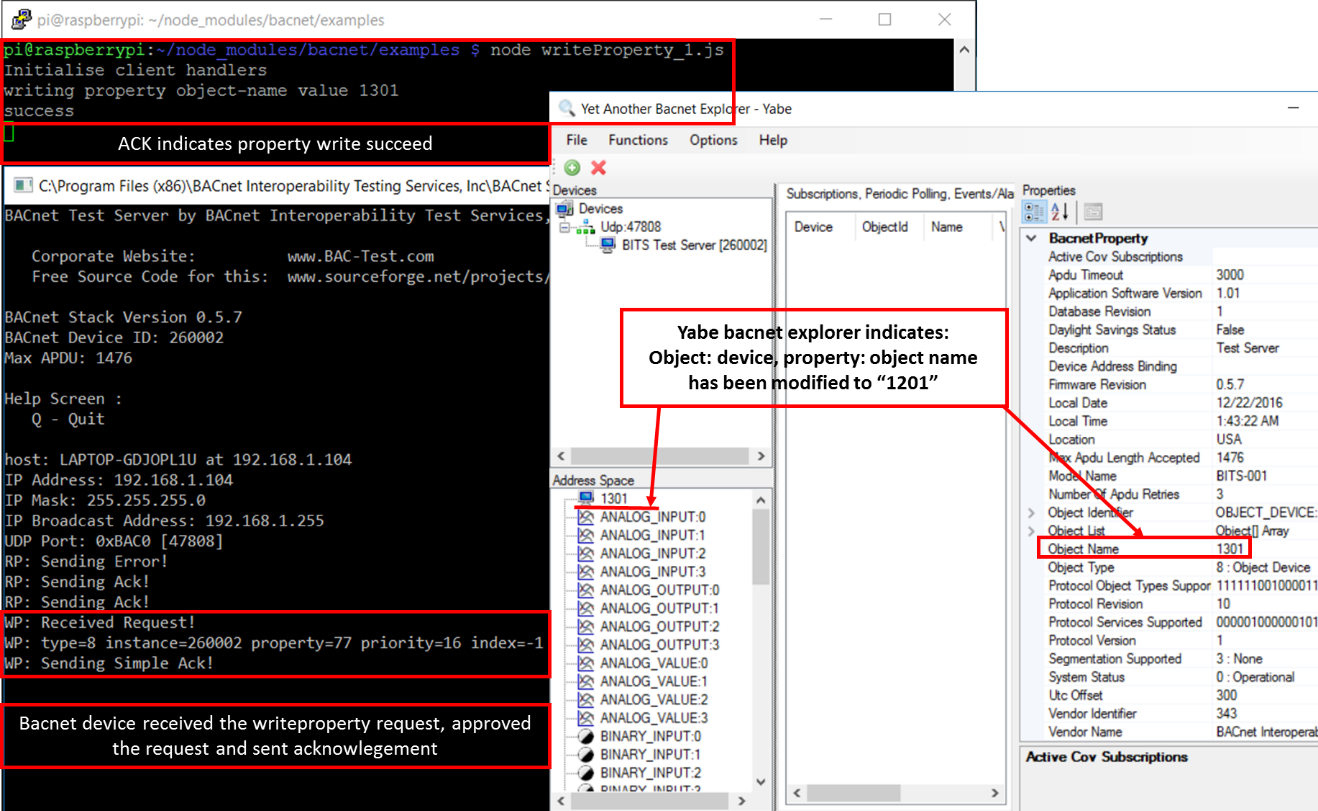
this prepares the property value that we want to write.

1. Target host ip address: xxx.xxx.xxx.xxx (for here it is 192.168.1.106).
2. Configure the arguments of r.writeProperty function

* Object: ‘device’ (currently, it can only be ‘device’)
* Object number: 260002
* Property (name): ‘object-name’
* Property value: variable value in json format.

After these modification, run this writeProperty.js script by:

$: node writeProperty.js



Create a block in node-red:

Execute node-red in the single board computer by

$: node-red

Open browser, enter xxxx.xxx.xxx.xxx:1880 (here 192.168.1.106:1880) to open the node-red interface,

Use unsafe-function (must be pre-installed) to create a node that has the writeproperty function.

Entering code as follows:

const bacnet = require('bacnet')

const r = bacnet.init({

datalink: {

iface: 'wlan0',

ip\_port: 0xBAC0

},

device: false

})

function withAddressOrId (addressOrId, callback) {

if (addressOrId.match(/^\d+$/)) { // integer - should be a device Id so we have to do a whois

console.log('doing whois for ' + addressOrId)

r.whois(Number(addressOrId))

r.on('iam', function (iam) {

console.log('iam: ', iam)

callback(Number(addressOrId))

})

} else { // something else - should be an address so we dont need to do a whois

callback(addressOrId)

}

}

//var value = JSON.parse(1)

var value =**'meem502**'

console.log('writing property', bacnet.propertyKeyToString('object-name'), 'value', value)

withAddressOrId('192.168.1.104', function (addressOrId) {

r.writeProperty('192.168.1.104','device',260002,'object-name', false, value, function (err) {

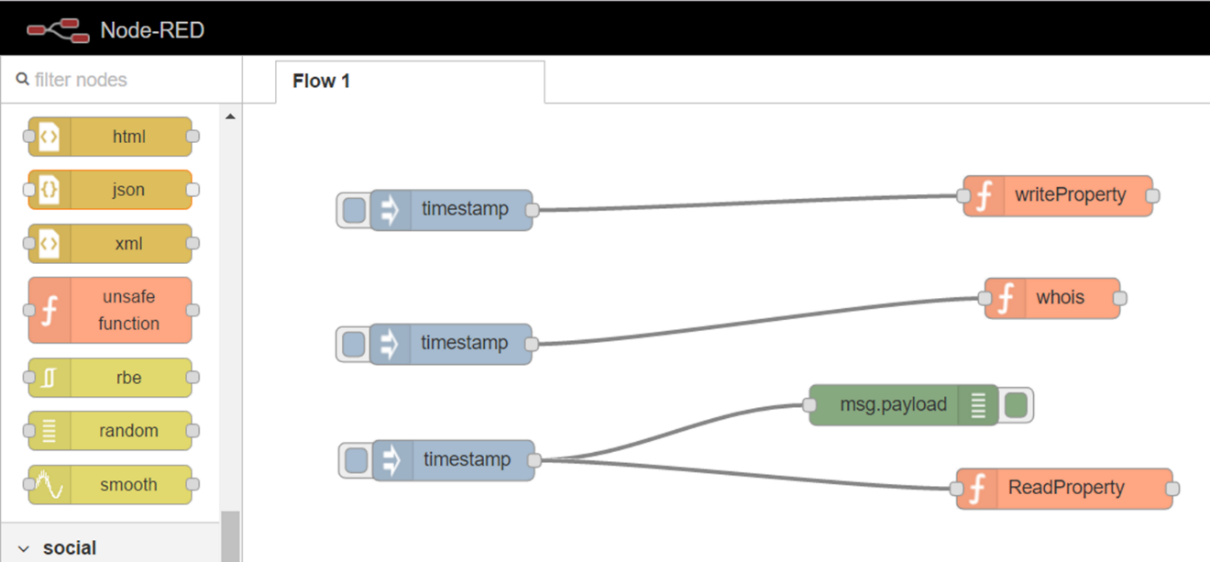
if (err) console.log('error', err)

else console.log('success')

})

})

setTimeout(function () {}, 1000)



The object-name of the ‘device’ object will be modified from “1301” to “meem502”

