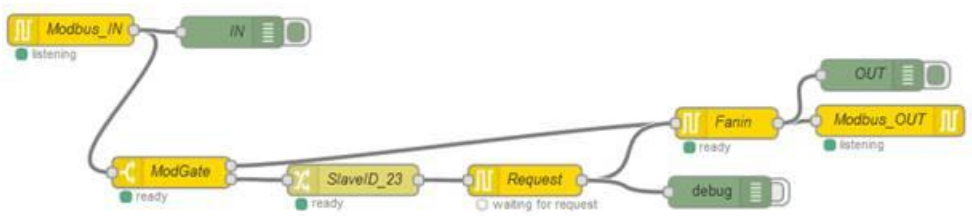
Modbus Example

Attached you can find two implementations on LuvitRED that will do what you want.

The first one, ModGate_1-1.json, is a direct 1-to-1 modbus gateway communication:



It basically has a modbus TCP (slave 1) receiving data and sending that data over the serial port 4 by changing the slave ID to 23, the response from slaveID 23 is send back to modbus TCP. In this case the registers of slave 1 are the same as the registers of the slave connected over the serial port.

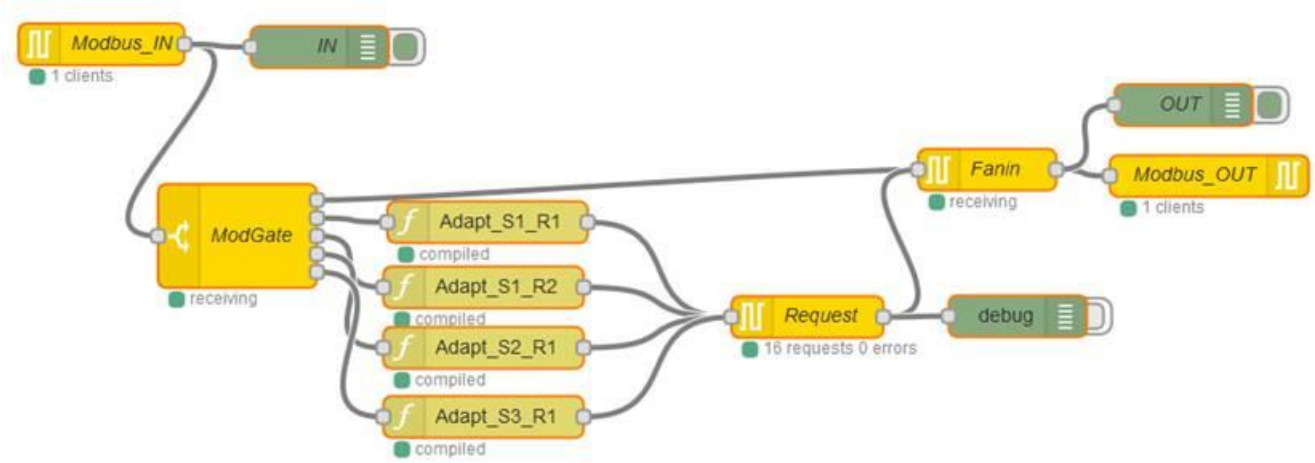
- **Modbus_IN** is the modbus TCP that is listening on port 502 (to remote access you need to add a firewall rule)
- **ModGate** is the place where the packages transmitted to Slave 1 Register X are send out to the second port (you could add mode mappings in this node). It is currently just passing every modbus command to the next node.
- **SlaveID_23** is performing the transformation on the original request so that it is mapped correctly to the right slave IDof the real modbus RTU device.
- **Request is** the node that actually sends the request to the modbus RTU device connected over port 4. The configuration for the port on the scenario I'm sending is: **Baud Rate: 9600, Data bits: 8, Parity: None, Stops Bits: 1**
- **Fanin** is a node that combines all the responses coming from Request and replaces all that is needed to be a correct response for the original request.
- **Modbus_OUT** is the modbus TCP out communication to send the response back to your request.

The second configuration I'm sending is a bit more complex, but it is just to give you an idea of what you can do with this setup:

The CloudGate (Modbus TCP slaveID 1) is receiving some requests and needs to retrieve the values from three different slaves and registers as per the following table:

CloudGate Reg	Slave	Register
1	1	1
2	1	2
3	2	1
4	3	1

So every CloudGate register is actually a register on a different slave over Modbus RTU. The scenario looks like this:



- **Modbus_IN** is the modbus TCP that is listening on port 502 (to remote access you need to add a firewall rule)
- **ModGate** is the place where the packages transmitted to Slave 1 Register X are send to the right output port. It is currently only accepting **Holding Registers**, if any of the values you want to get are not holding registers, then we need to change something on the configuration of this node.
- **Adapt_Sx_Rx** is a little transformation on the original request so that it is mapped correctly to the right slave ID and register on the real modbus RTU devices.
- **Request is** the node that actually sends the request to the modbus RTU device connected over port 4. The configuration for the port on the scenario I'm sending is: **Baud Rate: 9600, Data bits: 8, Parity: None, Stops Bits: 1**
- **Fanin** is a node that combines all the responses coming from Request and replaces all that is needed to be a correct response for the original request.
- **Modbus_OUT** is the modbus TCP out communication to send the response back to your request.

In order to get access over the WAN interface, you need to add the following firewall rules:

Inbound port forwarding

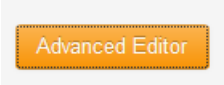
Protocol	Inbound interface	Source IP	Dest. port	Target IP : port	Actions
TCP	ALL	Any	8081	192.168.1.1:8081	
TCP	ALL	Any	502	192.168.1.1:502	

Add

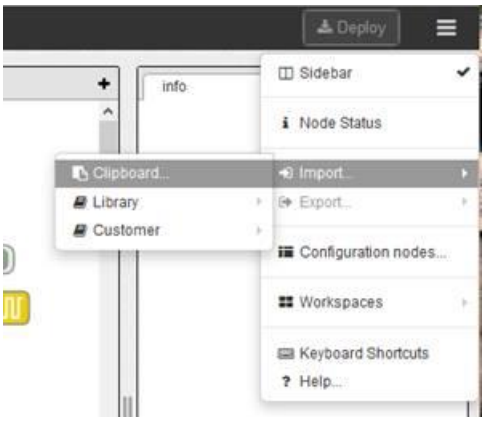
8081 for accessing LuvitRED remotely and 502 for the Modbus TCP communication.

In order to use the configurations:

1. Open one of the attached files on a text editor
2. Select and copy the contents
3. Open the advance editor of LuvitRED:

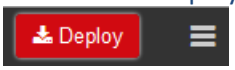


4. Go to the menu (top right), then Import from Clipboard:



5. Paste the contents of the file:

6. Drag the configuration to the middle
7. Click on a suitable location
8. Press on the Deploy button on the top right:



I hope this helps you understand how the ModGate configuration works.