Setup Without a NUC Image

Setup

1. Install Ubuntu 18.04.3 LTS
2. Setup the two accounts as described below.
3. Install the RT Patch (s.t. realtime programming becomes possible)
4. Setup NTP for time synchronisation
5. Install RTI DDS Setup for network communication, or **eProsima**. The file **install_eprosima_on_nuc.bash** in the software folder of the repository should contain all vital information for this, regarding required commit IDs for the branches to work with the branches chosen in the repository for thirdparty etc. You can also use this script directly to remotely install eProsima on one or multiple NUCs.
6. Install Matlab Setup (optional) and the RTI Matlab Support Package
7. Install the following packages: git, tree, openssh, tmux, cmake, libgtkmm-3.0-dev
8. (Optional) Install the raspbian toolchain and the Raspberry Pi RTI libraries
9. Make sure that you set up a guest account with a simple password. Generate a SSH Key and put the public Key in the LCC main computer, so that a log-in without using a password becomes possible
10. Enable the autostart script as mentioned above. The script is located in the software repository.
11. You also need to make sure that the autostart script is started after the NTP clock sync. This is done by another sudo-autostart script, which communicates with lab_autostart (see below).
12. You need to allow the guest user to restart the machine without "sudo". This is also explained below.
13. If desired, set some environment variables, e.g. the RTI location, the location of the Matlab executable etc
14. Setup the NUC's IP: 192.168.1.2xx, where xx corresponds to the NUC's ID ID 8 - 208, ID 15 - 215 etc.

Accounts

We use two user accounts on each NUC. Nevertheless, the setup can be used with only one user. The LCC will establish a connection to each NUC using `ssh ACCOUNT@IP-ADDRESS.`

Account: controller

The "root" account that can be used to install software or updates, change system settings etc. It is password-protected.

Account: guest

The account used for running the HLC scripts / software on the NUCs. For an SSH connection, no password is required as long as the local SSH key is installed on the current system. This connection is e.g. used by the Lab scripts to upload scripts, so make sure to use these scripts on a machine that owns the key. If it is not on your device, simply copy the key from the Lab's main computer: It is located in `home/USERNAME/.ssh/id-rsa` and should be copied to a similar location on the desired computer.

This account is also password-protected, but - if necessary - the password can be shared to use the account without the key-based SSH authentication.

Autostart

Enable autostart **for the guest user** (on which all remote deploy actions are perfomed), using `crontab -e` and insert

```
@reboot ~/autostart/lab_autostart.bash
```

This makes the HLC regularly send messages that it was turned on and allows the user to determine which NUCs are online. (Make sure that chmod +x was performed on the file before, and that it does not use relative file locations. Also, make sure that the directory exists and put the autostart script from software/hlc/autostart into the same folder. The autostart program (also from software) is downloaded from the apache web server using the autostart script, which runs on the lab's main computer - if the software has been built on it)

Domain ID: 21. Topic name: hlc_startup

In the given directory should be lab_autostart.bash as well as the executable of the autostart program which is called by the bash script

ⓘ lab_autostart.bash - if it ever gets changed - needs to be uploaded onto each NUC afterwards, to replace the older file.

ⓘ lab_autostart.bash uses lab_update.bash in a future update. If it is also part of the current repository, do not forget to put this file on the NUC as well!

ⓘ **Software out of date**

If the software on the NUCs is not up-to-date, check in /var/www/html/nuc on the main PC if all (four) packages downloaded by lab_autostart. bash on the NUCs are provided (and if they are up-to-date)

Upload not working

If the upload to the NUCs is not working, check that their fingerprint has already been accepted (needs to be done when no connection has been established before just manually connect via SSH to the NUCs on your PC once before using the LCC to deploy the files remotely)

NTP Sync Before Autostart

Use the controller account for this task. We will create a startup script that runs as sudo to enforce a NTP time sync before lab_autostart is started. This step is important, because all timers, including the one started in lab_autostart, rely on the system clock. If it is changed 'backwards' (into the past), the timer wakes up only when it reaches its next timestep set in the future when the clock had a higher value. This might cause it to stop its periodic execution for minutes up to hours, and is definitely undesired behavior.

The following script needs to be inserted. Read the comments above, as they explain where to put the script and how to activate it:

```
#!/bin/bash

# Place in /etc/rc.local
# The service only starts if the file is formatted properly, like this, with your code in between:

# Also, you need to make sure that the file is executable: sudo chmod +x /etc/rc.local

# Another problem: We want the lab / hlc program to only start if the time sync was finished
# This is done by using pipes between lab_autostart and the enforced time sync in rc.local

# For communication between processes, proposed solution:
# -------------------------------------------------------
# -------------------------------------------------------
# -------------------------------------------------------

# First, set a wrong date (for testing purposes)
# date +%T -s "14:14:00"

# Ping to make sure that an internet connection is available
# Write to /dev/null to suppress output
until ping -c1 ntp1.rwth-aachen.de >/dev/null 2>&1; do sleep 0.1; done

# Now enforce time sync
service ntp stop
ntpd -gq
service ntp restart

# Then, setup communication with the NUC startup script
# We could also run this as another user from here
# But that would be more complicated
# (The script needs to be run from guest)
# We tell the script that we are finished with the clock sync
# It is sufficient to just create a pipe here - we will check for its existence
# in the other script using -p
nuc_ntp_pipe=/tmp/nuc_ntp_pipe
nuc_lab_pipe=/tmp/nuc_lab_pipe

mkfifo /tmp/nuc_ntp_pipe
mkfifo /tmp/nuc_lab_pipe

chmod a+rwx /tmp/nuc_ntp_pipe
chmod a+rwx /tmp/nuc_lab_pipe

# delete the named pipe on exit
trap "rm $nuc_ntp_pipe $nuc_lab_pipe" EXIT

# Communication: 1 to tell that time sync is done, expect 1 as answer
echo "1" > $nuc_ntp_pipe
while read ans < $nuc_lab_pipe; do
 if [[ $ans -eq 1 ]]; then
 break
 fi
 sleep 1
done

# Also just for testing purposes
echo "Done" >> /tmp/test.txt

exit 0

# The answer is given by lab_autostart, which is already automatically started (see Documentation)
```

The autostart script needs to include the following lines after the export statements. They should already be part of the master branch, so you just need to make sure that the script you put on the NUC is up-to-date:

```
# -------------------------------- WAIT FOR TIME SYNC -------------------------------------------------------
# Wait for clock sync before doing anything else, because starting any program before a clock sync would cause
problems
# The clock sync is performed in rc.local of the sudo user
# We communicate via pipes, alternatively starting this script as another user would have been possible as well
# Check for existing pipe, wait until it has been created
nuc_ntp_pipe=/tmp/nuc_ntp_pipe
nuc_lab_pipe=/tmp/nuc_lab_pipe



while [[ ! (-p $nuc_ntp_pipe && -p $nuc_lab_pipe ) ]]; do
 sleep 1
done

# Read msg from pipe, which is sent after the time sync, then answer
# 1 -> time sync done, answer with 1 on other pipe
while read sy < $nuc_ntp_pipe; do
 if [[ $sy -eq 1 ]]; then
 break
 fi
 sleep 1
done
echo "1" > $nuc_lab_pipe
# ---------------------------------------------------------------------------------------------------------------
```

Restart / Sudo Change

Use the controller account for this task.

Type `sudo visudo`. Then append the following to the file:

```
# Guest user can reboot the system without password (for LCC reboot option)
guest ALL=NOPASSWD: /sbin/halt, /sbin/reboot, /sbin/poweroff
```

Repositories

- cpm library
- Lab software