TALLINN UNIVERSITY OF TECHNOLOGY
School of Information Technologies

Siim Soopalu 192516IASM

Smart home system framework selection

Master's thesis

Supervisor: Andres Rähni
MSc

Tallinn 2022

TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond

Siim Soopalu 192516IASM

Targa kodu süsteemi raamistiku valik

Magistritöö

Juhendaja: Andres Rähni
MSc

Tallinn 2022

Author’s declaration of originality

I hereby certify that I am the sole author of this thesis. All the used materials, references to the literature and the work of others have been referred to. This thesis has not been presented for examination anywhere else.

Author: Siim Soopalu

09.05.2022

Abstract

Smart home system developers might want to integrate devices of different brands and technologies into one system but using different apps for different brand devices can be cumbersome. That would also not allow integrating devices to automate the system for a more comfortable setting that a smart home system should provide. To bring various brands devices and different technologies together multiple platforms have been developed to allow home automation despite having different brand devices.

This work will describe and compare two of the most popular open-source home automation frameworks to help, average people wishing to create a smart home, choose the best suiting platform to use. Frameworks that the author used: Home Assistant, openHAB. This work will describe the experience of the author trying different frameworks to create a simple smart home system. To give more value, security risks of different wireless technologies were researched, possibilities that frameworks offer to increase smart home system security, and things smart home developers and users can do to secure their devices and network are outlined.

This thesis is written in English and is 46 pages long, including 5 chapters, 18 figures and 2tables.

Annotatsioon

Targa kodu süsteemi raamistiku valik

Targa kodu arendaja võib soovida kasutada erinvate tootjate seadmeid ja erinevaid tehnoloogiaid, millega seadmed ühilduvad. Selleks ei piisa tootjate oma arendatud äppidesse seadmete ühendamisest, kuna need ei paku integratsiooni võimalust teiste tootjate seadmetega ega automatiseerimise võimalust. Seda probleemi lahendavad kodu automatiseerimise raamistikud, mida on viimase 10 aasta jooksul tekkinud mitmeid. Käesoleva töö eesmärk on võrrelda kolme erinevat raamistikku ning valida välja sobivaim, keskmisele inimesele, kes soovib oma kodu targemaks teha.

Töö vältel kirjeldatakse kahte populaarsemat platvormi Home Assistant ja openHAB ning võrreldakse autori kogemusest neid kasutades lihtsa targa kodu süsteemi üles seadmisel. Lisaväärtuse pakkumiseks uuriti erinevate juhtmevaba tehnoloogiate turvalisust, raamistike poolt pakutavaid võimalusi süsteemi turvalisemaks muutmiseks ning mida saab targa kodu arendaja ja kasutaja saavad teha, et süsteem turvaline hoida.

Lõputöö on kirjutatud inglise keeles ning sisaldab teksti 46 leheküljel, 5 peatükki, 18 joonist, 2 tabelit.

List of abbreviations and terms

IoT	Internet of Things
UI	User interface
HA	Home Assistant
OS	Operating system
openHAB	open Home Automation Bus
OSGi	Open Services Gateway initiative
PC	Personal computer
GB	Gigabyte
RAM	Random access memory
MCU	Microcontroller unit
A	Ampere
V	Volt
AC	Alternating current
Hz	Hertz
kPa	Kilopascal
WPA	Wi-Fi protected access
HTTPS	Hypertext Transfer Protocol Secure
HTTP	Hypertext Transfer Protocol
CSA	Connectivity Standards Alliance
MFA	Multi-Factor Authentication
MQTT	Message Queuing Telemetry Transport

Table of Contents

List of figures

List of tables

1 Introduction

IoT (Internet of Things) devices being sold annually keeps growing every year, for example in 2019 there were 10 billion IoT devices sold, in 2020 11.3 billion, forecasts for 2021 were that there would be 12.3 billion IoT devices sold. That makes around 10% growth annually, but that may change because of the ongoing digitalization of industries and the rise of popularity in smart homes. Forecasts of IoT devices sold in 2025 will be more than 27 billion [1]. It was estimated that households will have 50 connected devices in 2021, including smartphones, laptops, speakers, and other devices [2]. That may seem like a lot of devices for one house but when building a smart home number of smart devices connected may increase very fast. Integrating devices, especially when they use different technologies, to make a reliable and comfortably automated smart home can become tricky and that is issue that needs attention.

1.1 Problem

Attempting to build a smart home system can be quite frustrating in terms of finding suitable devices to fit the requirements of the system and specially to find them from the same the manufacturer because often manufacturers develop phone applications to work with only their devices. Furthermore, these applications developed by smart device manufacturers do not include possibilities to integrate different services a user might like and capabilities to customize the layout of the UI (user interface) or create automations and rules to automate the smart home. This issue has been around for a long time and some people have taken up the challenge to integrate different manufacturers devices, creating automations and using different web-based services to be able to create a desirable smart home system.

IoT devices used in smart home applications are using wireless technology like Wi-Fi, Bluetooth, Zigbee or other technologies. These can be easy targets to malicious attacks of home networks when the devices and networks are not protected well. Keeping a home network safe is a must because 75% of IoT attacks are from infected routers but that does

not mean that every IoT device for sale is secure and could be trusted to add to any network and system because reportedly malware found in IoT devices is growing 30% each year [3].

1.2 Goal

Goal of the current thesis is to compare popular smart home automation frameworks by going through a practical task of creating a simple system with selected devices and evaluating which has a bigger upside. As wireless devices and networks may carry big security risks, this thesis will also cover security risks of more popular wireless technologies, framework specific security features and what can the user do to minimize the risk of getting their smart home hacked.

1.3 Structure

Remaining work is separated into four chapters. Chapter 2 gives a brief overview of two smart home development frameworks, what hardware platforms they can be used on and in short try to summarize how many and what kind of devices these platforms support.

Following this, chapter 3 will summarize the practical work of building a sample smart home system with chosen devices and trying to add automations that can be useful in real life using different frameworks and comparing the experience. This also includes comparing the features available in building an UI. Lastly this chapter will evaluate the frameworks based on different categories to find out which framework has more upsides for the user.

Second to last chapter will talk about vulnerabilities of different wireless technologies used in smart home devices and how they can be mitigated. Also, there will be a summary of each frameworks available features that help increase the systems security. In the last section of this chapter, this thesis will provide some instructions to users on what they can do to have a more secure home network.

Finally, the last chapter will be summarized.

2 Framework comparison

This chapter will briefly introduce different smart home developing frameworks and an overview of what hardware it is required to run them. Also, an overview of what these frameworks are capable of in terms of devices available in them with some examples.

2.1 Framework description

2.1.1 Home Assistant

HA (Home Assistant) was founded in 2013 by Paulus Schoutsen [4]. It is an open-source software for home automation focusing on privacy and local control [5]. It gives users the opportunity to run the smart home server locally on a desktop, Raspberry Pi, Android, IOS or other platforms [6], instead of running it in cloud which may lead to several issues. HA is developed in Python, one of the most popular coding languages in the world which means there are many contributors contributing their work to this project [7].

HA OS (operating system) is supported on several hardware platforms and for each an installation tutorial can be found with firmware repository links. Here is a list of hardware HA is supported on [6]:

- Raspberry Pi – Support for Raspberry Pi 3 and 4.
- ODROID – ODROID N2+ is recommended but ODROID N2, C2, C4, and XU4 are supported also.
- ASUS Tinkerboard
- Generic x86-64 – PC (Personal computer) with AMD64 or Intel64 processor, for example Intel NUC. Must be exclusively available for HA OS
- Windows, macOS, Linux – Will run in the virtual machine, image is provided by HA.

HA has gained lots of users over the years and are trying to offer support for users by having a forum or as they call it a community, where people can post about issues or ideas of different topics and others can share their thoughts and help [8]. On top of that there is

a Discord chatroom for HA users with over 90 000 users [9]. For the people who wish to contribute to HA there is a webpage dedicated for developers on how to use API-s (application programming interface) or for example develop add-ons [10].

2.1.2 OpenHAB

The openHAB (open Home Automation Bus), founded by Kai Kreuzer in 2010 [11], is an open-source home automation software that is meant to run locally, but can use some cloud services, like HA and in principle is very similar to HA [12]. Biggest difference between the two is that openHAB is developed in Java. OpenHab software allows the user to integrate all the different devices and technologies supported to one solution with the features available in the software. This means that there is no limit to different manufacturer devices and technologies used as long they are supported by openHAB.

OpenHAB uses OSGi (Open Services Gateway initiative) for modularity, which is developed by the OSGi Alliance, a non-profit organization founded in March 1999. OSGi allows developers to use modules developed with specific standards to create vendor-independent, standards-based approach to modularizing Java software applications and infrastructure [13].

There are different platforms that openHAB support running the software [14]:

- Linux, Windows, macOS – No restrictions on Linux version noted, Java 11 JVM (Java Virtual Machine) must be installed on the system. Recommended is Zulu.

- Raspberry Pi – Quite famous platform for openHAB, recommended is Raspberry Pi 4 with 2 or 4 GB RAM, 3 A power supply and 16 GB SD (secure digital) card, preferably a card that allows more write cycles than an average SD card to better endure openHAB's use conditions. If Raspberry Pi 4 with over 2 GB RAM is not available, minimum requirement is to use a Raspberry Pi 2, 3 or 4 with 1 GB or more.

- Armbian – openHAB will run on any Armbian supported hardware, list of hardware can be found on Armbian website [15], with minimum requirements of:

    - 512 Kb (kilobyte) of memory

    - eMMC or a quality SD card

    - 4 cores
    - Wired networking
- Docker
- Synology DiskStation – Although Synology NAS (Network-attached storage) is based on Linux it has limitations, not being a full server. DSM 7.0 and higher is supported.

2.2 Support of devices and services

2.2.1 HA

To build a smart home system in HA users can use integrations and add-ons. These allow adding devices and using or integrating different services with them, helping add different capabilities like using your voice to turn the kitchen lights on or visualization of energy usage.

Integrations are supported by the HA community and during the writing of this thesis there are 1963 of them [16]. The number of integrations available does not show how many devices are supported. Mostly it means what brand devices are supported and within that brand integration can be many devices supported.

Add-ons are available in the add-on store which can be accessed when running HA. There are 3 types of Add-ons:

1) Official add-ons seen in figure 1 [16].
2) HA Community add-ons, which are developed and maintained by only members of the HA Community and can be found in GitHub and GitLab [17] [18].
3) Third-party add-ons, which can be created by anyone, but must be used on the users own risk because HA cannot guarantee the quality and security of these [19]. Third-party add-ons can be imported from repositories, for example: GitHub, GitLab.

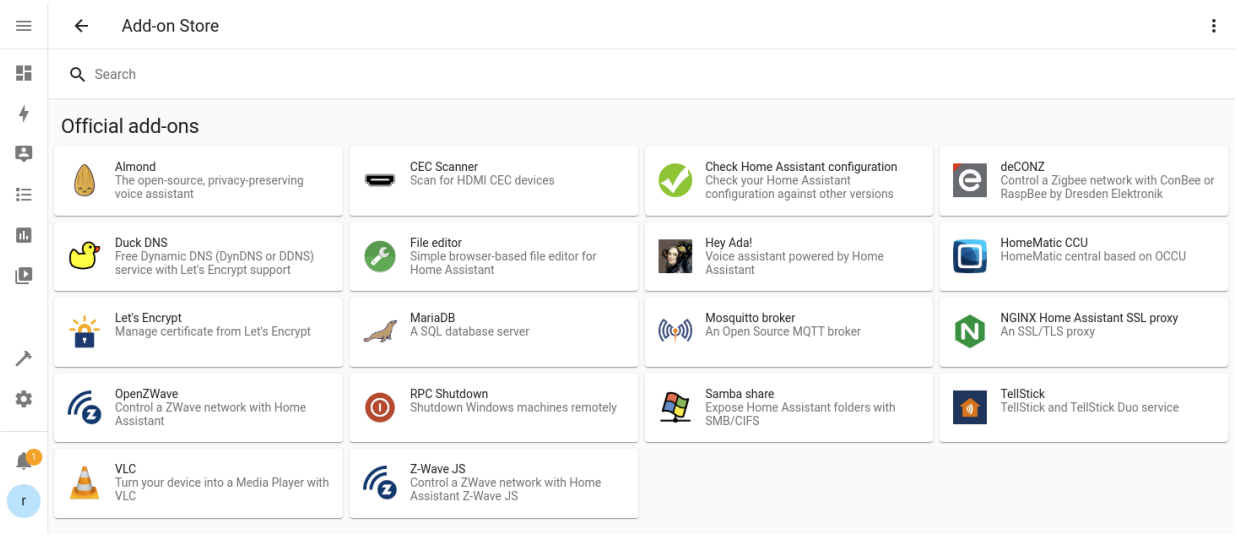


Figure 1. Official add-ons in Add-on Store

List seen below shows some of the well-known brands supported [16].

- Nest – Google developed smart devices, found in following categories: binary sensor, camera, climate, doorbell, hub, media source, sensor.
- Apple TV – Allows control of any generation Apple TV, found in following categories: media player, multimedia, remote.
- Coinbase – Lets access account balances and exchange rates, found in following categories: finance, sensor.
- GitHub – Allows to monitor favorite repositories, found in sensor category.
- Tesla Powerwall – Allows to monitor different states and parameters of the Powerwall, found in following categories: binary sensor, sensor.
- Acer projector – Allows control of RS232 connected projectors, found in multimedia category.

2.2.2 OpenHAB

To build smart home systems using openHAB users can use add-ons, also called bindings, and things to create powerful and easy to use systems that automates everyday life tasks and makes living at home more comfortably.

In total openHAB has 370 add-ons which are mostly official and accepted add-ons maintained by openHAB maintainers found in GitHub [20]. Most featured add-ons are displayed in figure 2 [21].

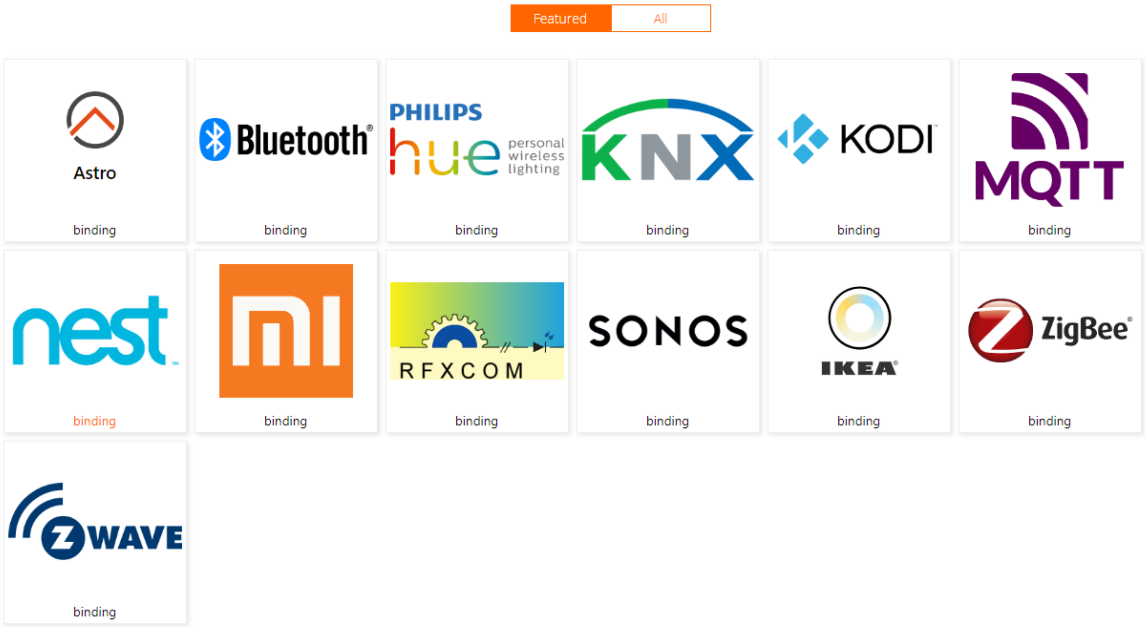


Figure 2. OpenHAB featured add-ons.

Things represent devices in openHAB and all the add-ons all together support 2854 things at the time of the writing of this thesis [21].

List seen below shows some of the well-known brands supported [21].

- Epson – Allows to connect to projectors that have built-in networking, communication happens via TCP or USB serial.
- Samsung TV – Allows control over specific Samsung TVs.
- Miele @home – Integrates Miele @home appliances equipped with required communication modules. Devices communicate through Zigbee and Wi-Fi.
- SolarEdge – Allows to retrieve live data from SolarEdge inverters.
- Yamaha – Allows control of specific Yamaha receivers.

3 System build

This section describes the practical part of this thesis about building a smart home system with an overview of installation of a framework on selected hardware system, selection of devices and the experience of setting them up, adding rules to automate the system, and creating a user interface in different smart home development frameworks.

3.1 Devices

Devices were selected keeping in mind that the system should be a realistic and find practical use but also try out more than 1 technology. Although the devices selected compose a simplistic system they are of different categories: switches, sensor, Zigbee hub, thermostat.

Below are outlined the devices used in the smart home system this thesis is based on:

Figure 3. Sonoff BASICR3 WIFI Smart switch

2 Sonoff BASICR3 WIFI Smart switches used in sample system, illustration seen in figure 3 [22]. This device works on ESP8285 MCU (microcontroller unit), can withstand maximum load of 10A (ampere) and works with 100-240V (volt) AC (alternating current) 50/60Hz (hertz) [23].

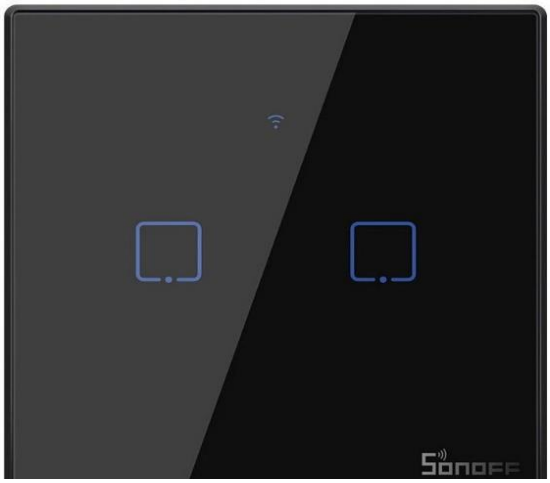


Figure 4. Sonoff TX series Wi-Fi light switch

1 Sonoff T3 2 channel Wi-Fi light switch with 2 channels seen in figure 4 [24]. This device works on ESP8285 MCU with maximum current output of 2A per channel and works with 100-240V AC 50/60Hz [25]. When device was flashed with Tasmota firmware and used in both frameworks, it often stated booting and got stuck in the boot loop and could not control it. When looping one of the lights was also

turning on and off, the other was not and likely because of the module was reset to default as one channel switch which had to be changed back to two channels when it go out of the booting loop. Unplugging the power to the device for some times often helped, but not everytime.

Figure 5. Dresden Elektronik Conbee II

1 Dresden Elektronik Conbee II seen in figure 5 [26]. Incorporates a power-amplifier to create a stronger signal and is compatible with many lights, switches and sensors from Philips Hue, IKEA TRÅDFRI, Xiaomi Aqara, OSRAM SMART+, Busch-Jaeger, GIRA, JUNG, Paulmann and Paul Neuhaus. Helps computers to create universal Zigbee gateways [26].

Figure 6. Aqara Temperature and Humidity sensor.

1 Aqara temperature and humidity sensor Zigbee seen in figure 6 [27]. Also monitors atmospheric pressure. Advertised maximum errors are temperature ±0.3°Celsius, humidity ±3%, and pressure ±0.12 kPa (kilopascal) with range 30 kPa – 110 kPa [27].

Figure 7. Beok Tuya Wi-Fi water floor heating thermostat.

1 Beok Tuya Wi-Fi water floor heating thermostat seen in figure 7 [28].

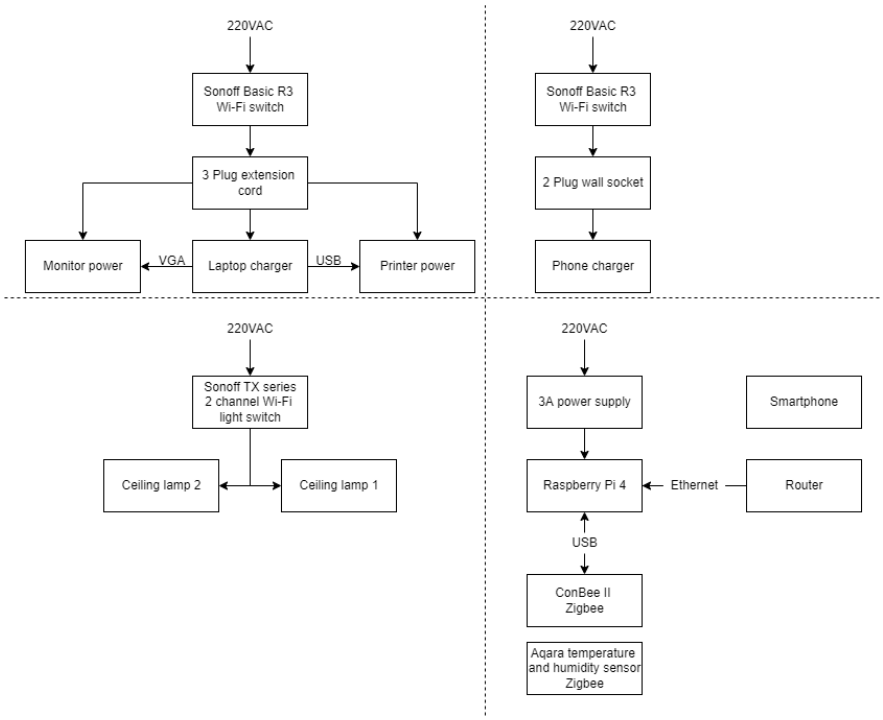


Figure 8. Illustration of system setup.

Illustration of devices used and assembled as a whole system can be found in figure 8. Sonoff Wi-Fi switches are setup to allow powering multiple devices, in this case one powers a 3-plug extension cord allowing to control power to the monitor, laptop and printer allowing to limit power draw of these devices when the user is not using them or leaves for an extended period. Other Sonoff Wi-Fi switch powers a 2-plug wall socket allowing to control power of two items, in this case only one was used to power the phone charger. This setup allows control of the phone charging based on phone battery level, which the authors phone does not support.

3.1.1 Tasmota firmware

From the available devices Sonoff devices can be flashed with the Tasmota firmware to give users easier control, via HTTP, MQTT and KNX [29], and more flexibility using the device. It is an open-source firmware for ESP8266, ESP32, ESP32-S or ESP32-C3 chipset-based devices [30] that can be integrated with many different home automation solutions like Home Assistant, openHAB, KNX [29]. Users can connect other devices to the available pins of the device they are using with Tasmota firmware, because the free pins on the device allow using GPIO, ADC, I^2C, serial, SPI peripherals to also integrate things like RTC, motor controllers, different sensors [31].

Tasmota offers a guide [32] on how to flash the software which requires a serial-to-USB-adapter, in this case FTDI FT232 was used. Next the 3.3V, GND (ground) and the serial communication pins of the adapter and the device need to be connected, for this the device needs to disassembled to access the pins on the device. To flash the firmware on the device four flashing tools are mentioned in the guide: Tasmota Web Installer, Tasmotizer, ESP-Flasher and Esptool.py. The first tool is used during this work. Next the device to be flashed with new firmware needs to be put into programming mode, to achieve this GPIO0 of the MCU must be connected to GND while the chip is booting. Putting the device in programming mode may vary based on the device, some devices have a button for that, others may need a jumper wire to do it. After flashing the Tasmota firmware on the device it will create a hotspot network which the user has to connect to configure the Wi-Fi network the device will be connected to. After that the user is directed to the IP address of the device which is ready to be controlled as seen in figure 9 for example.

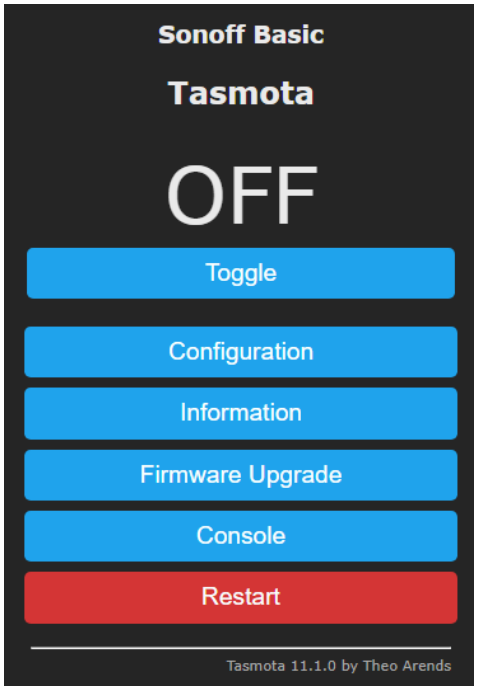


Figure 9. Sonoff basic flashed with Tasmota HTTP page.

3.2 Workflow of connecting devices

3.2.1 HA

Installing HA OS on the Raspberry Pi and getting it running was straight forward following the instructions and after a successful installation, account creation and login the next step was to get all devices connected to HA.

Starting with the Sonoff devices, there are two options: through the eWeLink app on a phone and syncing with a third-party eWeLink add-on or flashing Tasmota firmware on the devices and connecting using MQTT (Message Queuing Telemetry Transport) and official Tasmota add-on. Trying the first option connecting all the devices with HA was quite easy but when trying to add automations to them it was not very straight forward, as the app itself was a device and the real devices were entities. Flashing the Tasmota

firmware on the Sonoff device was quite easy and it will be covered in detail in section. Connecting Sonoff devices using Tasmota firmware with HA was also simple, if help is needed there are a lot of tutorials on Youtube. Mosquitto broker add-on needed to be installed, MQTT integration added, and all devices had to be configured to connect to the MQTT broker. After that they could be easily added to the UI and make some automations.

Connecting the Aqara Temperature and Humidity sensor to HA did not require much effort using the deConz add-on and integration and no instruction were needed as every step was simple to understand. Device itself took a long time to get connected to the Conbee II stick but that is not the fault of HA. Adding the device to the UI was simple as there was a device instance of the sensor available.

Adding the Tuya smart thermostat to HA could be done using the Tuya integration which connects to Tuya IoT cloud where a person must create an IoT project where their devices will be added. To add the device to the cloud project user must have Tuya app installed on their phone from where they can connect to the device in question. To connect the app and the cloud project, the cloud generates a QR-code that the phone app must scan, in this case the phone's rear camera had a focusing problem and could not scan the code preventing connecting the smart thermostat to HA. Tuya phone app does not have a feature to use the front camera. Because of this problem the thermostat will not be used in the HA system. The documentation provided by HA to use Tuya integration was easy to understand and get to that point of scanning the code.

3.2.2 OpenHAB

Like with HA, installing openHAB on the Raspberry Pi and getting it running was straight forward following the instructions and after a successful installation, account creation and login the next step was to get all devices connected to openHAB.

First device connected to openHAB was Aqara Temperature and Humidity sensor with the help of deConz binding that was necessary to be installed, but the binding itself does not support Raspbee and Conbee sticks. To use Conbee II stick in openHAB openHABian Configuration Tool had to be accessed from the command line console to install deCONZ companion app for Conbee/Raspbee sticks under optional components section. Accessing the command line console was achieved via SSH using PuTTY. After successful optional

deCONZ app installation a port was assigned to access Phoscon web app. Connecting the sensor to the Zigbee stick had no issues. After refreshing the openHAB page the connected device was finally found and displayed as three (temperature, humidity, pressure) separate pre-configured things ready to be added to the things list. To add these things to the UI to display values first a channel needed to be chosen for each thing, then linked to an item which also created a model. Finally, all the selected values to be displayed could be added to the UI.

Next, Sonoff devices with Tasmota firmware were added to openHAB using MQTT which meant that Mosquitto needed to be installed from the Configuration Tools optional components section. Also, MQTT Binding needed to be installed to create the MQTT broker and devices to be connected things. After configuring the brokers hostname/IP and port the Sonoff devices MQTT parameters needed to be configured from the devices IP address to get them connected to the created MQTT things. From here the process of getting the control of the device to the UI was like what was done before with the multi sensor things. The process of getting the Sonoff devices connected to openHAB took around 3 hours, the devices were not able to connect to the MQTT broker, but the restart of openHAB seemed to fix it. Sometimes clicking the mouse cursor on items did not function as if it were disabled, refreshing the page seemed to fix that.

Regarding the Tuya thermostat, openHAB does not have a Tuya specific binding meaning that this device is not directly supported in this framework. Tasmota does not support this Tuya device also. Although searching help openHAB community showed that a GitHub repository tuya-mqtt could be installed from the command line console. Cloning the repository took some time but finally it was cloned but the steps after cloning got more difficult and harder to understand from community posts so the attempt to connect the Tuya thermostat failed again.

3.3 Rules

In this section, the author attempts to create four automations, described in the list below using both frameworks with the simple setup, seen in figure 8.

1. Sonoff switch controlling the extension cord will be powered on when arriving home and powered off when leaving home.

2. Sonoff switch controlling the wall socket will be powered on when phone battery drops below 20% and keep it on until it reaches 85%. When reaching 85% power is switched off and will not be turned on before reaching 20%.

3. Sonoff light switch will turn on both lights when arriving home between 20.00 and 05.00.

4. Three notifications will be sent to the smartphone through openHAB app based on temperature and humidity sensors values: 1) “Getting hot!” when temperature value rises over 26℃, 2) “Getting cold!” when temperature value falls below 20℃, 3) “Getting wet!” when humidity value rises over 50%.

3.3.1 HA

HA automations consist of triggers, conditions, and actions out of which conditions are optional. Each of them can be of many different types and multiple instances. Below are outlined the process and experience of creating automations listed above using HA.

When installing the HA application on the smartphone and logging into HA, which does not require cloud, different parameters of the smartphone are automatically added to the entities list to use like battery level, battery temperature, is charging or not and if location access is allowed also location is shared. Installing the app and accessing the HA home server was simple.

Creating the first automation went well and took about 2 minutes to set up. There are two triggers: when the phone enters or leaves a chosen zone and two actions: first action is based on a condition that the phone is not home and the power is turned off, second action is the default action of turning the power on.

Automation two is triggered by change of the battery level above 0% and below 100%, values can be defined between which values it triggers, but cannot be left empty. There are two conditional actions and one default, seen in figure 10 and 11. When conditions are not met, the default action is performed.

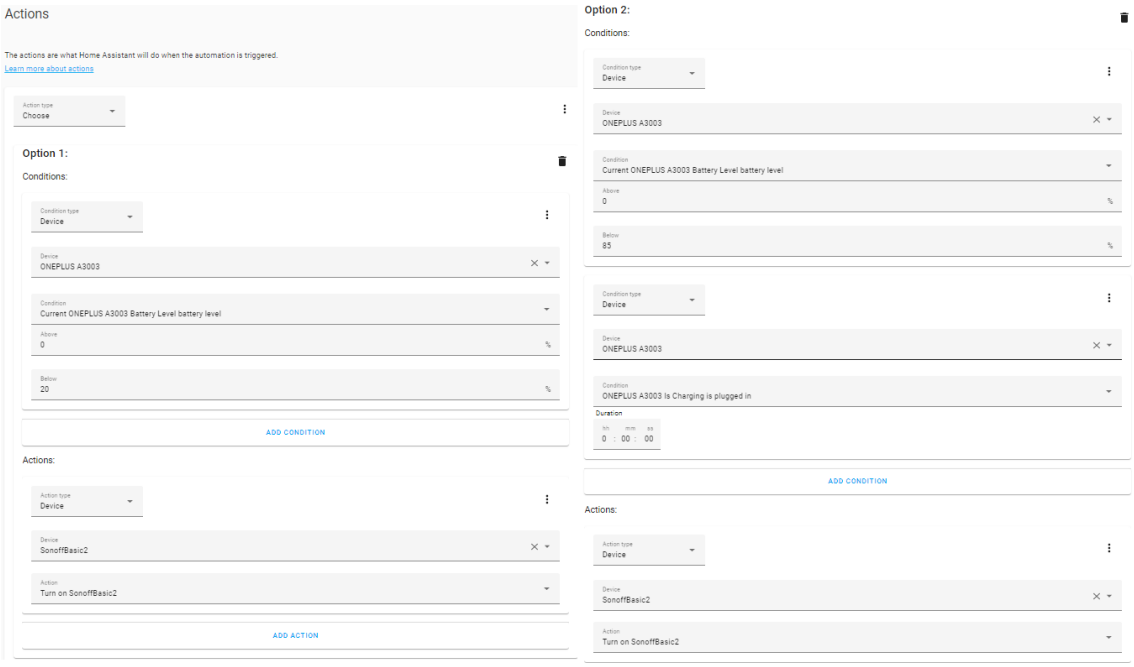


Figure 10. Automation 2 option 1 action on the left, option 2 action on the right, HA.

First condition is when battery is below 20% send command to turn power on, second condition is when battery is below 85% but still charging, keep the power on and the default action, when first two conditions are false, turns the power off. This automation took around 10 minutes to complete.

Figure 11. Default action in automation 2 HA.

Creating automation 3 was very straightforward, it is triggered by phone entering specified zone to send command to turn both lights on but only after 20.00 and before 05.00. Two actions had to be added, one for both light switch channel.

Easiest way to implement automation 4 was to create two separate automations, one based on temperature and the other based on humidity. Ability to send push notifications to the phone app is already present in services under developer settings, it became available when logging in HA server from the phone app, like the other parameters mentioned in the second paragraph of this section. Also, integer helpers were needed to save previous

values of temperature and humidity which were done as actions in corresponding automations. The process of saving a value to a helper was quite difficult when not knowing how it is done, even documentation did not show examples of that. Automation regarding the temperature was triggered by the change of temperature value followed by two actions. First action is chosen from two options, first condition is that if temperature is higher than 26℃ and the previous value “Getting hot!” is sent as the notification but when temperature is lower than 20℃ and the previous value “Getting cold!” is send as the notification. The second action performed is saving the current value so it could next time be used to compared to.

3.3.2 OpenHAB

OpenHAB allows automations through rules that start with a trigger and end with an action which can also be configured to happen only when some conditions are met. A rule can have multiple triggers, actions, and conditions.

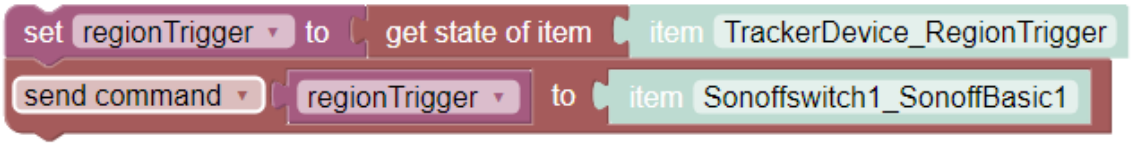


Figure 12. Rule 1 openHAB Blockly script.

To get updates if a person has left home GPSTracker binding was installed and connected with the OwnTracker app, over HTTP, that had to be installed on the smartphone. Through this app location coordinates, GPS accuracy, timestamp of last report, phone battery level and region trigger can be used. Latter was used to get information if person was in the configured radius of set region coordinates or not and that had to be configured in the phone app. This rules trigger was that when region trigger value changed a script

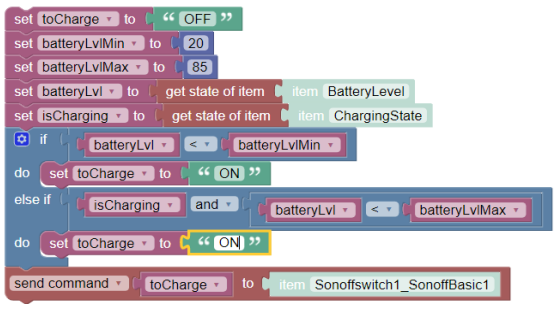


Figure 13. Rule 2 openHAB Blockly script

```
var toCharge, batteryLvlMin, batteryLvlMax, batteryLvl, isCharging;


toCharge = 'OFF';
batteryLvlMin = 20;
batteryLvlMax = 85;
batteryLvl = itemRegistry.getItem('BatteryLevel').getState();
isCharging = itemRegistry.getItem('ChargingState').getState();
if (batteryLvl < batteryLvlMin) {
  toCharge = 'ON';
} else if (isCharging && batteryLvl < batteryLvlMax) {
  toCharge = 'ON';
}
events.sendCommand('Sonoffswitch1_SonoffBasic1', toCharge);

```

Figure 14.Rule 2 openHAB Blockly generated code.

will be run made with Blockly, seen in figure 12. Setting this rule up went mostly without problems and was done in about 30 minutes.

Although the GPSTracker binding working with OwnTracker provides the battery level of the phone it is not enough for the second automation because it also requires the information if the phone is currently charging, otherwise after reaching 85% and dropping to 84% the switch turns on again. Because of that another approach was needed. OpenHAB phone app provides the possibility to share some information about the phone with the framework but only through cloud. For that an account had to be created in myopenhab.org to connect to openHAB remotely and the openHAB Cloud binding had to be installed. To register the account an UUID and a secret key was needed that was generated with the installation of the openHAB Cloud binding but could only be reached in the local files using the command line console. Next step was to enable sending device information in the openHAB phone app and creating items for battery level and charging state in openHAB that the values are attached to. The rule for this automation is triggered by the change of the battery level that triggers a script made with Blockly, seen in figure 13 and the code generated by Blockly can be figure 14. Setting up this rule 2 took the most time because getting openHAB cloud in myopenhab.org working with openHAB had many setbacks for the author: figuring out where to find the UUID and secret key, finding how to send battery level from openHAB phone app and lastly getting the script to work, it can be quite tricky for a person who has never tried scripting like this. In total implementing rule 2 took about 4 hours.

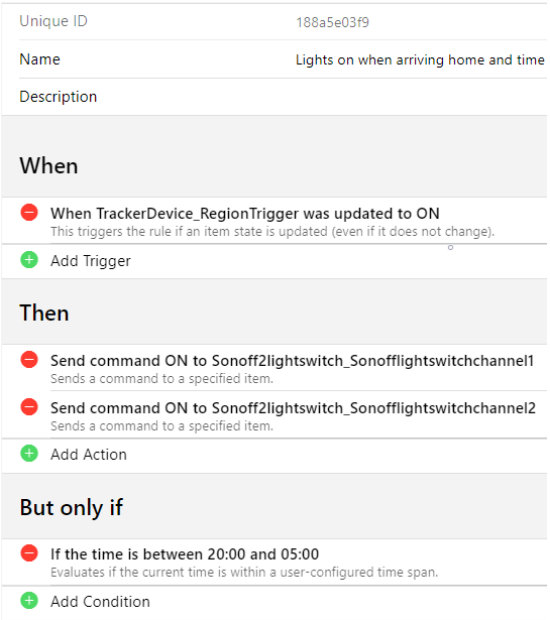
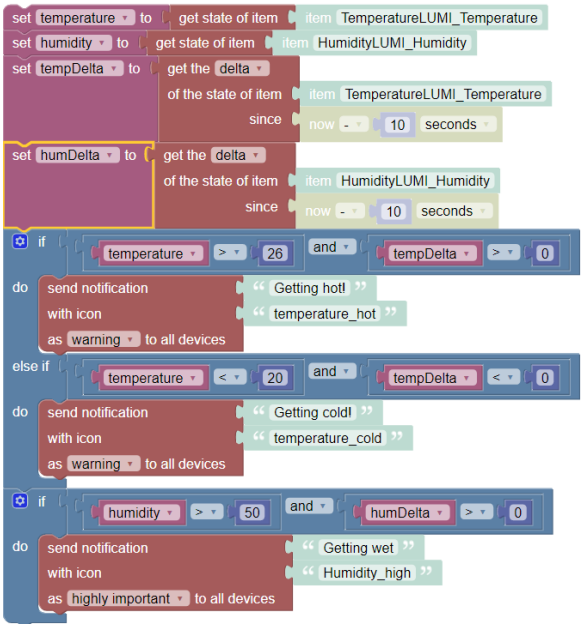


Figure 15. Rule 3 in openHAB

Rule 3 requires the region trigger also used in rule 1 which when updated to “ON” sends “ON” command to both channels of the light switch only when the time is between 20:00 and 05:00. The whole rule can be seen in figure 1. Setting up rule 3 was simple and took about 1 minute.



Rule 4 is triggered by the change of either the humidity value or the temperature value which in turn will start a script made with Blockly, seen in figure 16, where temperature and humidity delta between the current value and the value ten seconds ago are used in conditional statements to decide whether notifications should be sent or not. Trickiest part of getting rule 4 to work properly was not getting more than one notification from each condition: too hot, too cold, or too wet.

Figure 16. Rule 4 in openHAB made with Blockly.

3.4 UI

3.4.1 HA

UI in HA is called a dashboard and tabs could be added there, which in HA terms means views. HA provides 29 different cards, each with different configuration options, which interface with the dashboard to display information but in addition to the cards provided by HA they have added additional resources in their documentation [33] where users can import custom cards. Below are outlined some of the HA provided cards [33]:

- Energy Cards – There are different cards to choose from that are involved in energy monitoring for example: energy usage graph, solar production graph, energy distribution, see in figure 17 [33].

- Glance Card – Useful to make a compact overview of different sensors, possibility to make it dynamic using the entity-filter, which allows to track entities only when in a certain state.

- Map Card – Displays entities on a map.

- Conditional Card – Displays different Cards based on entity states.

- Webpage Card – Allows to integrate users favourite webpage into HA but cannot add sites using HTTP if the user uses HTTPS for HA.

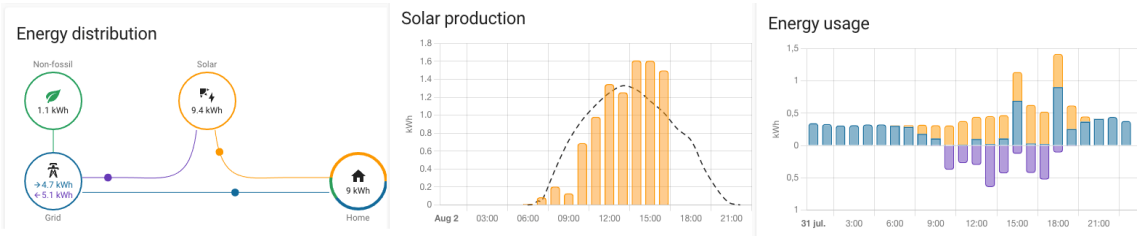


Figure 17. 3 Energy cards: on the left Energy distribution card, middle Solar production, right Energy usage.

3.4.2 OpenHAB

With the openHAB3 release in December 2020 openHAB tried to modernize and make creating UI-s more user friendly [34]. Semantic modelling was introduced already in previous versions but with the new release there is much more use of this feature, letting openHAB understand what the systems items are representing – if it is a location such as a room, physical object, or an interaction point [34]. In the beginning using semantic models can be intimidating and cumbersome but later it could make the developers life much easier and develop faster be it creating rules or creating an UI.

Main term used in building UI in openHAB is pages and there are several types of them listed and described below [35]:

- Home Page – Consists of 4 tabs: Overview, Location, Equipment and Properties. When building the semantic model these tabs will be filled based on the model but can be configured by the developer if needed.

- Sitemaps – Consists of hierarchies of pages and sub-pages that have simple controls in them. They offer less customization options than other types, making them the easiest to learn. Main web UI is not able to display them at the time of writing this work. Sitemaps are meant to be used in mobile applications for Android and iOS.

- Layout Pages – Introduced in openHAB3 offering the most versatile option of displaying information in the main UI. Layouts have many options to control how things are laid out and can display widgets customized from built-in libraries or

from the developer's personal library, that may include designed or imported widgets.

- Maps & Floorplans – Offer a background image overlayed with markers or other elements. Maps have map layer or a satellite view from a provider service. Floorplans have a custom image provided by the user.

- Charts – Provide interactive visualization of persistent data with a selectable period and date to show. User can create custom charts or save the analyzer chart of an items value like temperature or humidity.

- Tabbed Pages – Allows combining other pages rendering them into tabs.

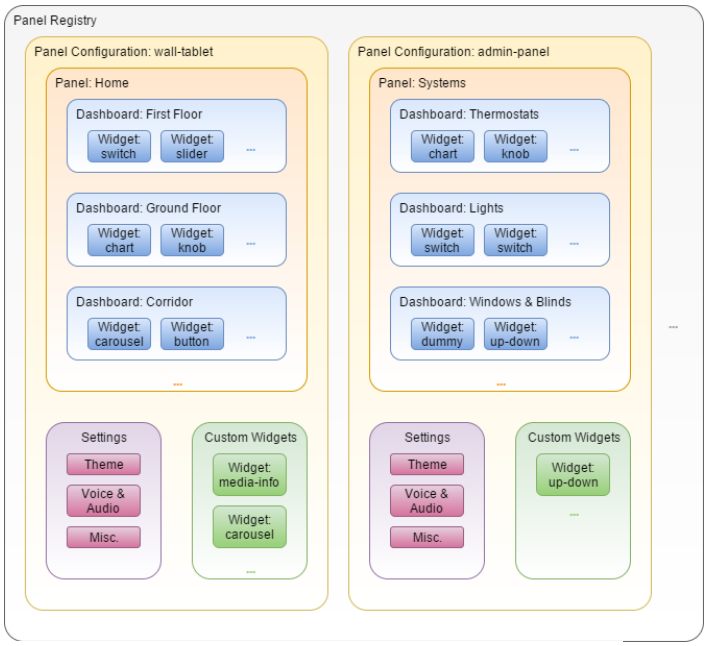


Figure 18. HABPanel architecture and entities.

As an alternative to sitemaps openHAB provides a feature called HABPanel UI suited for example wall-mounted tablets. Although sitemap and HABPanel are similar, they are different concepts and can be designed independently because they are not related. HABPanel has its own terminology, seen in figure 18 [36]and briefly described below [36]:

- Panel Registry – Contains several Panel Configurations.

- Panel Configuration – Container holding a Panel with its Settings and defined Widgets.

- Panel – Set of dashboards that can be presented to end users, who can switch between them using the menu.

- Dashboard – Comprises of widgets.

3.5 Evalution

Final part of this chapter will summarize the author's experience of using and findings made while working with HA and openHAB and make a verdict which would in the author's opinion be a better platform to use for an average tinkerer.

3.5.1 Ease of use

Out of the two frameworks used HA was the easier and more intuitive to use. OpenHAB does confuse with the abundance of terminology and configurability, it is easy to make a mistake and get stuck for a long period of time trying to troubleshoot what went wrong. Especially when the software itself does not act as it is intended, like when clicking on the mouse cursor does not do anything and refreshing the page only fixes it or accidentally closing the sidebar and cannot find a way to get it back is fixed with a system restart. Although HA feels more intuitive and easier to use it still needs getting used to.

3.5.2 Automations

When creating automations with the visual editors HA has more flexibility with things to use as triggers or actions and using YAML (yet another markup language) adds a little bit to that. Although OpenHAB visual editor does not give as many options than HA it provides possibilities of scripting logic in Blockly which is quite simple to understand and more complex develop logic. Scripting with Javascript can be used to create powerful automations in openHAB.

3.5.3 UI

When comparing flexibility and configurability creating UI, openHAB has the advantage specially when semantic modelling is used effort can be reduced but it comes at a cost of ease of use and understandability at least for users. Building an UI in HA is a much simpler task, but the outcome may not be as good as it could be with openHAB.

3.5.4 Support

Both frameworks have communities where people post their issues or ideas and help others, but the rate of post made in both communities are very different. At the time of writing counting post made in last 30 minutes shows that in HA there were 13 made in the last 30 minutes but in openHAB community there was 1 [8] [37].

Often people search for practical videos from Youtube if they are having issues and availability for each frameworks tutorials on different topics cannot be determined with Youtube search because the number of videos found is not displayed there. Instead, another option was tried. When using Google search engine the keyword Home Assistant for videos the results were 257 000 000 but with the keyword openHAB 102 000 videos were found. Although they are all not Youtube videos, it illustrates that HA has far more support and tutorial videos than openHAB.

3.5.5 Overall

Both frameworks have their advantages over the other, but to summarize the author's experience the conclusion is that HA is a great option for people new to home automation because openHAB is quite difficult to grasp for people unfamiliar with the framework, but OpenHAB is a great option to advanced users.

4 Network and protocol security

IoT is helping companies, in different industries, automate manufacturing, data transfer and other processes by integrating or adding smart devices to their existing systems. Most of the smart devices work over wireless protocols and may pose security risks. Same goes for smart homes that are using smart devices to automate actions in the home setting to make lives of people living there more comfortably. Smart devices use different wireless protocols like Wi-Fi, Bluetooth, Zigbee, and Z-Wave. In table 1 [38] specifications of Wi-Fi, Zigbee and Z-Wave are outlined which provide users guideline in choosing the wireless technology of a device they wish to add to their smart home system. This topic will cover what network security risks each of these protocols have, what smart home developers and users can do to minimize these risks.

Table 1. Comparison of wireless technology specifications.

Wireless Technology	**Wi-Fi**	**Zigbee**	**Z-Wave**
Operating Frequency	2.4 GHz 5 GHz	868/915 MHz, 2.4 GHz	908.42 MHz in US, 868.42MHz in EU
Maximum Data Rate	54 Mb/s	250 kbit/s	40 or 100 kbit/s
Nominal Range	100 m	10 m	~30 m indoor
Power Consumption	~116 mA	~ 40 mA	~ 2.5 mA

4.1 Protocol security

4.1.1 Wi-Fi

Wi-Fi, also called wireless connectivity, was invented in 1997 and introduced for home use in 1999 [39]. This technology allows to connect PC, laptop, tablets, mobile phones to connect at high speed to the internet without the need for a physical wired connection. Wi-Fi is the radio signal sent from a wireless router to a nearby device, The device transmits a radio signal back to the router, which connects to the internet by wire or cable [40]. In table 2 [7] are shown types of Wi-Fi connections with their advantages and disadvantages outlined.

Table 2. Types of Wi-Fi connections.

	Advantages	Disadvantages
Wireline/router	• Convenient setup. • Connect multiple devices.	• Limited bandwidth. • Speed reduces with more devices connected. • Might not be accessible in rural areas.
Mobile hotspot	• Any smartphone or tablet can be used as a hotspot device. • Can be used on the go.	• Drains a lot of battery. • Uses data fast.
Jetpack hotspot	• Can be used on the go. • More devices can connect to it than smartphone. • Greater range than smartphone.	• Buying separate plan than smartphone from mobile carrier.

4G LTE	• Available in rural areas. • Faster than hotspot.	• Availability may be limited in some places. • Cost of service and setup.
5G	• Cost-efficient. • Faster than 4G. • Lower latency than 4G.	• Limited availability.

Although, Wi-Fi has existed for over 20 years now and during this time security of Wi-Fi has been improved and tested, some weaknesses have been discovered.

In 2017 researchers from DistriNet Research Group, Mathy Vanoef and Frank Piessens, released a paper about how attackers within range of a victim could exploit WPA2 (Wi-Fi protected access) 2 protocol weaknesses using key reinstallation attacks [41]. All protected Wi-Fi networks use a 4-way handshake, which is a process of exchanging 4 messages between the supplicant and the authenticator to generate encryption keys that are used to encrypt data sent over Wi-Fi [41]. They already have a shared Pairwise Master key and during the handshake a fresh session key called the Pairwise Transient Key is negotiated. The 4-way handshake was thought to be secure for 14 years [41] until M.Vanhoef and F.Piessens found this issue by targeting these handshakes, WPA and WPA2 certified products are affected by their attacks [41]. They discovered similar weaknesses in other Wi-Fi handshakes like the PeerKey handshake, the group key handshake, and the Fast BSS Transition handshake.

The good news is that this problem, for many devices, can be fixed with an update, the bad news is that the update must be made by the vendor which may never come but there is a list found in GitHub about different vendors and their progress with the patch [42]. Also, M.Vanhoef has shared a repository in GitHub of scripts that people can run on their device to test if clients or access points are affected by the KRACK attack against WPA2 [43].

In 2021 researcher M.Vanhoef released a paper about finding 3 design flaws in the 802.11 standard which affects all protected Wi-Fi networks ranging from WEP (Wired Equivalent Privacy) to WPA3, meaning that these flaws have been part of Wi-Fi since its release in 1997 [44]. One of the flaws is the 802.11's frame aggregation functionality and other two in the 802.11's frame fragmentation feature [44]. M.Vanhoef found that on top the 3 design flaws in the Wi-Fi standard there were widespread programming mistakes in Wi-Fi products and through experiments confirmed that every Wi-Fi product is affected by at least one, but most are affected by several vulnerabilities [45].

These security risks can be mitigated by using HTTPS (Hypertext Transfer Protocol Secure) websites and keeping you network router firmware up to date with the latest updates [45]. Author of the forementioned discovery has created a tool which can test Wi-Fi clients and access points for the 3 discovered design flaws, but only some wireless network cards are supported [46].

4.1.2 Zigbee

ZigBee is a wireless protocol developed by CSA (Connectivity Standards Alliance), before rebranding called ZigBee alliance, which consists of 247 semiconductor and software developing companies like Logitech, Mitsubishi Electric, Mastercard, Nordic semiconductor, ON semiconductor and many more [47]. CSA provides certification tools and packages for manufacturers to certify their Zigbee products to guarantee interoperability with other Zigbee devices and give credibility of the device. They also share testing providers on their website to help with the certification process.

Zigbee based smart devices are low power, cost and data rate compared to Wi-Fi or Bluetooth which make them popular in smart home applications because most smart home devices do not require high data rate, like switches, sensors, and thermostats [48]. Zigbee devices need to be connected to a trust centre, also called a hub or router which and they can compose a mesh network. Also, devices communicate to the smart home server, be it a PC, Raspberry Pi, or something else, through the hub because these servers do not have the capabilities to understand Zigbee protocol.

Zigbee protocol itself provides a secure environment for example by using 128-bit Advanced Encryption Standard (AES) to encrypt and authenticate messages but it does not mean that the Zigbee product itself is secure. Level of security found in a Zigbee

device is a choice of the product manufacturer. Main security risks in a Zigbee wireless network are [49]:

- Theft of sensitive data from a node – This can be user data or network security data, like encryption keys, that allow access to the node and network. Effort must be made to avoid eavesdropping as well as re-join and replay attacks, but most critical is to protect network keys.

- Theft of a node – A node can be removed and moved to another network where it can be accessed and controlled. This is a great risk when a node is added to the network through Touchlink commissioning but can be mitigated by stopping the device from responding to Touchlink messages from other networks and disabling Touchlink on the device and allowing re-enabling manually or by authorized person.

- Unauthorized control of a node – May result from theft of sensitive data, a node or replay attacks. Fortunately, Zigbee security feature of frame counters help defeat replay attacks and changing the network key regularly can also help.

- Loss of network service – This may happen by jamming of the radio channel or entire radio band by interference. Named problem can be mitigated by allowing applications to change used frequency by moving the network to another channel.

The main cause of forementioned vulnerabilities is the network key which is the weakest security element of the Zigbee network. It is used by all nodes in the network and when a node joins the network it is passed encrypted with a pre-configured link key, this link key is commonly of the ‘global’ type and open for exposure. There is no mandate how link keys should be distributed and is up to each vendor to use a mechanism, they see fit, for securing the keys. This can be prevented by programming a random install code, consisting of 6, 8, 12 or 16 bytes with an additional 2-byte CRC (Cyclic redundancy check), in the Zigbee device in the factory which will be used to derive a link key from the Zigbee stack [49].

4.1.3 Z-Wave

Today Z-Wave is developed by the Z-Wave Alliance that comprises of many industry leaders throughout the world with a goal to bring practical wireless products and services

to market by any brand or vendor. Some known members of this alliance are Silicon Labs, Assa Abloy, Danfoss, Sharp [50]. In principal Z-Wave is quite similar to Zigbee by being developed by an alliance to make the protocol interoperable with other brand devices using Z-Wave. Also, it uses considerably less power than Wi-Fi but has even lower data rate than Zigbee. Like Zigbee, Z-Wave also requires a centre hub for devices to connect to.

In a paper, published in 2020 by School of Cybersecurity in South Korea, is shown that attacks carried out on real-world Z-Wave devices can cause damage to smart home residents [51]. Below list illustrates just a few issues found in Z-Wave technology and does not include all vulnerabilities of this technology.

- Downgrade attacks – Z-Wave chips have different levels of pairing security process, S0, S1 and S2. S2 being the most secure. These chips can be downgraded from S2 to S0 by an attacker, being present at the time of the pairing of a device. This allows the attacker to intercept the network key, giving them access and possibility to inject S0 traffic to the whole Z-Wave network. Only mitigation technique at this point seems to be to make sure there is nobody, who should not be near the pairing process, in the range of the Z-Wave network when pairing a device [52].

- DoS (Denial of Service) attack – Attacker intends to make the network inaccessible to users. Effectively preventable by disabling frame segmentation feature in a Z-Wave device which is not used in practical applications anyway [51].

- Beware of devices using Silicon Labs chipsets – There are 10 results for the keyword Z-Wave (CVE Search). 2 out of 10 are general Z-Wave vulnerabilities and 5 are about Silicon Labs chipsets.

4.2 Framework safety, security features

4.2.1 HA

Anyone can scan the entire web finding billions of websites out of which many are peoples HA instances which can be hacked into if proper precautions are not taken.

During the writing of this paper there are over 17 000 that a website called Shodan could find [53]. HA provides options to make an instance harder to find and if it is found there are measures that can be taken to make sure nobody gets into the HA instance which are outlined in the following list [54].

- MFA (Multi-Factor Authentication) – HA provides this feature to create a second security layer to passwords this can be used. Time-based one-time password module is the available MFA in HA.

- Limit access attempts – By default hackers can try to log into a person's HA instance as much as they like, but by enabling this IP (internet protocol) filtering and banning can be used to block login attempts after set number of times of failed logins. It would be recommended to add the automation of getting notifications of failed login attempts.

- Do not use hosting files – Static files can be stored when creating a directory called www under the configuration path /config. Although this option is developed by HA, it is not recommended because it can be accessed by anyone by not even authenticating themselves.

- Keep up to date – Update everything: OS, HA, add-ons, devices, anything associated with the instance or network. Vulnerabilities are found every day and often fixes for this are developed frequently.

- Encrypt traffic – When not encrypting traffic anyone with access can see it in plain text, including passwords. DuckDNS add-on is one way to encrypt traffic of a HA instance.

4.2.2 OpenHAB

Like HA instances, openHAB instance can also be visible in the web with over 2000 visible searching with Shodan [55]. Below is a list of actions openHAB allows to take to secure access to openHAB [14].

- Encrypted communication – Recommended to use secure HTTPS port instead of HTTP. By default, HTTPS port is 8443 and HTTP port is 8080, but these can be changed in configurations.

- Authentication – Possibility to limit access to certain network interfaces.

4.3 What can the user do for security?

Previous chapters have shown that security of a smart home device is mostly up to the developer of the device, but security risks of a smart home system are not limited to devices used in that system. Also, the home Wi-Fi network can have many vulnerabilities if measures are not taken to create a more secure home network. Here are some ways to add more security to a Wi-Fi network [56]:

- Change router default name – By default routers name is usually its make and model, with that information it is easy to look up the default login and password.

- Strong passwords – Set a unique password made up of letters, numbers and symbols or use a random password generator. To add more security also change the password with some time interval. This goes not only about the router, but also other passwords that are required for your smart home system.

- Multilevel authentication – Adds a second layer of security to the log in process. This can be in a form of verification code to phone or email, third-party applications like Google Authenticator.

- Highest encryption levels – The higher the better, WPA3 is currently the highest, but WPA2 is also good. If the router being used for the home network supports WPA or WEP protocols, then it might be a good idea to upgrade to another router.

- Separate network for smart home devices – If possible, use the feature to create a separate network dedicated to IoT devices. In case a IoT device gets hacked, the devices in the main network are affected.

- Keep things up to date – With updates not only do the features get better but often security is upgraded also, so it is a good idea to update IoT devices and home router as soon as possible.

- VPN (Virtual private network) – Allows access to a smart home network from anywhere with an internet connection through a private and secure, encrypted network. This can help hide instance or home network from the web.

5 Summary

During this work two of the most popular home automation frameworks, HA and openHAB, were used in creating a simple smart home system to compare them and find out what they are capable of and if on is better than the other. This work also intended to find out what kind of security risks different wireless technologies used in home automation have and if anything could be done about it.

Both tested home automation frameworks in general fill the same purpose and look the same until they are used, then the differences come out. The sample system for this work consisted of six devices from 4 different brands using two different technologies. Both systems could support them and managed to connect all but one device, which was the smart thermostat that needed adding to the framework through a phone application communicating to the cloud. Weak spot was not the device itself or the framework, but the damaged phone camera needed to scan a QR-code to connect the app with the cloud. There were also issues with Sonoff light switch that sometimes started boot looping and the author had no control over it.

Important distinctions made between the two used frameworks through the author's experience were that overall, HA is the easier framework to use due to being more intuitive and having a lot of support from the HA Community or other tutorials and videos. OpenHAB is harder to understand because of the many terminologies and extent of configurability that may lead to confusion and errors or bugs but offers more possibilities designing an UI or scripting complex automations.

All wireless technologies have security issues even today after tens of years from their birth but most important thing to know when buying wireless smart devices is that the brand is responsible for making a device as secure as possible. Things to keep wireless networks in homes secure are creating multiple security layers between a network and hackers like creating unique passwords, which are recommended to change after some period, and MFA. It is up to each person to make an effort to keep their networks and devices safe.

6 References

[1] S. Sinha, “State of IoT 2021: Number of connected IoT devices growing 9% to 12.3 billion globally, cellular IoT now surpassing 2 billion,” IoT Analytics, 22 September 2021. [Online]. Available: https://iot-analytics.com/number-connected-iot-devices/. [Accessed 3 May 2022].

[2] J. Steward, “The Ultimate List of Internet of Things Statistics for 2022,” Findstack, 15 February 2022. [Online]. Available: https://findstack.com/internet-of-things-statistics/. [Accessed 3 May 2022].

[3] M. Prokopets, “Hacking Statistics,” Nira, [Online]. Available: https://nira.com/hacking-statistics/. [Accessed 3 May 2022].

[4] LinkedIn, “Paulus Scoutsen,” LinkedIn, [Online]. Available: https://www.linkedin.com/in/schoutsen/. [Accessed 14 April 2022].

[5] Home Assistant, “Home Assistant,” Home Assistant, [Online]. Available: https://www.home-assistant.io/. [Accessed 20 April 2022].

[6] Home Assistant, “Installation,” Home Assistant, [Online]. Available: https://www.home-assistant.io/installation/. [Accessed 2 May 2022].

[7] Micheal_Davydov, “HomeAssistant for newcomers: What it is, what is hassio, hassos, hassbian, 101 and cookies,” Home Assistant, June 2019. [Online]. Available: https://community.home-assistant.io/t/homeassistant-for-newcomers-what-it-is-what-is-hassio-hassos-hassbian-101-and-cookies/123004. [Accessed 14 April 2022].

[8] Home Assistant, “Community,” Home Assistant, [Online]. Available: https://community.home-assistant.io/. [Accessed 2 May 2022].

[9] Discord, “Home Assistant,” Discord, [Online]. Available: https://discord.com/invite/home-assistant. [Accessed 2 May 2022].

[10] Home Assistant, “Developer Docs,” Home Assistant, [Online]. Available: https://developers.home-assistant.io/. [Accessed 2 May 2022].

[11] K. Kreuzer, “About me,” [Online]. Available: https://www.kaikreuzer.de/about/. [Accessed 15 April 2022].

[12] openHAB, “Introduction,” openHAB, [Online]. Available: https://www.openhab.org/docs/. [Accessed 28 April 2022].

[13] OSGi, “What Is OSGi?,” OSGi Working Group, [Online]. Available: https://www.osgi.org/resources/what-is-osgi/. [Accessed 3 May 2022].

[14] openHAB, “Installation,” openHAB, [Online]. Available: https://www.openhab.org/docs/installation/. [Accessed 3 May 2022].

[15] Armbian, “Download,” Armbian, [Online]. Available: https://www.armbian.com/download/?device_support=Supported. [Accessed 3 May 2022].

[16] Home Assistant, “Integrations,” Home Assistant, [Online]. Available: https://www.home-assistant.io/integrations/. [Accessed 3 May 2022].

[17] hassio-addons, “Repository,” GitHub, [Online]. Available: https://github.com/hassio-addons/repository. [Accessed 3 May 2022].

[18] hassio-addons, “repository,” GitLab, [Online]. Available: https://gitlab.com/hassio-addons/repository. [Accessed 3 May 2022].

[19] Home Assistant, “Common Tasks - Operating System,” Home Assistant, [Online]. Available: https://www.home-assistant.io/common-tasks/os#installing-third-party-add-ons. [Accessed 3 May 2022].

[20] openhab, “openhab-addons,” GitHub, [Online]. Available: https://github.com/openhab/openhab-addons. [Accessed 2 May 2022].

[21] openHAB, “Add-on Reference,” openHAB, [Online]. Available: https://www.openhab.org/addons/. [Accessed 3 May 2022].

[22] Sonoff, “BASICR3 Product Documents,” Sonoff, [Online]. Available: https://sonoff.tech/product-document/diy-smart-switch-doc/basicr3-doc/. [Accessed 1 May 2022].

[23] Sonoff, “BASICR3/RFR3 Product Specification,” 18 December 2020. [Online]. Available: https://sonoff.tech/wp-content/uploads/2021/06/%E4%BA%A7%E5%93%81%E5%8F%82%E6%95%B0%E8%A1%A8-BASICR3-RFR3-20201218.pdf. [Accessed 7 May 2022].

[24] Tasmota, “Sonoff TX T3 EU 2 Gang Switch(T3EU2C),” Tasmota, [Online]. Available: https://templates.blakadder.com/sonoff_TX-T3EU2C.html. [Accessed 3 May 2022].

[25] Sonoff, “TX EU Product Specification,” 16 November 2020. [Online]. Available: https://sonoff.tech/wp-content/uploads/2021/03/%E4%BA%A7%E5%93%81%E5%8F%82%E6%95%B0%E8%A1%A8-TX-EU-20201116.pdf. [Accessed 7 May 2022].

[26] Phoscon, “Conbee II,” Phoscon, [Online]. Available: https://phoscon.de/en/conbee2. [Accessed 1 May 2022].

[27] Aqara, “Aqara Temperature and Humidity Sensor,” Aqara, [Online]. Available: https://www.aqara.com/us/temperature_humidity_sensor.html. [Accessed 28 April 2022].

[28] Amazon, “Beok Tuya Smart Thermostat Heating Thermostat Room Thermostat WiFi Thermostat Intelligent Wall Thermostat for Water Heating Underfloor Heating Compatible Alexa Google 3A TDR83WIFI,” Amazon, [Online]. Available: https://www.amazon.de/gp/product/B09CGTQ3VD/ref=ppx_yo_dt_b_asin_title_o01_s00?ie=UTF8&th=1. [Accessed 28 April 2022].

[29] M. Salerno, “What is Tasmota and What Can it do for You?,” IoTRant, 18 October 2018. [Online]. Available: https://iotrant.com/2018/10/18/what-is-tasmota-and-what-can-it-do-for-you/. [Accessed 7 May 2022].

[30] Tasmota, “About,” Tasmota, [Online]. Available: https://tasmota.github.io/docs/About/. [Accessed 7 May 2022].

[31] Tasmota, “Peripherals,” Tasmota, [Online]. Available: https://tasmota.github.io/docs/Supported-Peripherals/. [Accessed 7 May 2022].

[32] Tasmota, “Getting Started,” Tasmota, [Online]. Available: https://tasmota.github.io/docs/Getting-Started/. [Accessed 7 May 2022].

[33] Home Assistant, “Dashboards,” Home Assistant, [Online]. Available: https://www.home-assistant.io/dashboards/. [Accessed 9 May 2022].

[34] openHAB, “openHAB 3.0 Release,” openHAB, 21 December 2020. [Online]. Available: https://www.openhab.org/blog/2020-12-21-openhab-3-0-release.html. [Accessed 9 May 2022].

[35] openHAB, “User Interface Design Overview,” openHAB, [Online]. Available: https://www.openhab.org/docs/ui/. [Accessed 9 May 2022].

[36] openHAB, “HABPanel,” openHAB, [Online]. Available: https://www.openhab.org/docs/configuration/habpanel.html. [Accessed 9 May 2022].

[37] openHAB, “Community,” openHAB, [Online]. Available: https://community.openhab.org/. [Accessed 9 May 2022].

[38] A. B. A. Rahman, “Comparison of Internet of Things (IoT) Data Link Protocols,” Washington University in St. Louis, St. Louis, 2015.

[39] J. Thomas, “The History of WiFi,” Purple, 25 May 2014. [Online]. Available: https://purple.ai/blogs/history-wifi/. [Accessed 28 April 2022].

[40] Verizon, “Wi-Fi Definiton,” Verizon, [Online]. Available: https://www.verizon.com/info/definitions/wifi/. [Accessed 28 April 2022].

[41] M. Vanhoef and P. Frank, “Key Reinstallation Attacks: Forcing Nonce Reuse in WPA2,” Leuven, 2017.

[42] kristate, “krackinfo,” GitHub, [Online]. Available: https://github.com/kristate/krackinfo. [Accessed 30 April 2022].

[43] vanhoefm, “krackattacks-scripts,” Gihub, [Online]. Available: https://github.com/vanhoefm/krackattacks-scripts. [Accessed 30 April 2022].

[44] M. Vanhoef, “Fragment and Forge: Breaking Wi-Fi Through Frame Aggregation and Fragmentation,” New York University, Abu Dhabi, 2021.

[45] M. Vanhoef, “Fragattack,” 11 May 2021. [Online]. Available: https://www.fragattacks.com/. [Accessed 28 April 2022].

[46] vanhoefm, “fragattacks,” GitHub, [Online]. Available: https://github.com/vanhoefm/fragattacks. [Accessed 30 April 2022].

[47] Connectivity Standards Alliance, “Participants,” Connectivity Standards Alliance, [Online]. Available: https://csa-iot.org/members/participants/. [Accessed 1 May 2022].

[48] Connectivity Standards Alliance, “Zigbee,” Connectivity Standards Alliance, [Online]. Available: https://csa-iot.org/all-solutions/zigbee/. [Accessed 1 May 2022].

[49] NXP Laboratories UK, “Maximizing Security in Zigbee Networks,” 2017.

[50] ZWave Alliance, “Member Companies of the Z-Wave Alliance,” ZWave Alliance, [Online]. Available: https://z-wavealliance.org/z-wave_alliance_member_companies/. [Accessed 1 May 2022].

[51] K. Kim, K. Cho, J. Lim, Y. H. Jung, M. S. Sung, S. B. Kim and H. K. Kim, “What’s your protocol: Vulnerabilities and security threats related to Z-Wave protocol,” School of Cybersecurity, Korea University, Seoul, 2020.

[52] A. Tierney, “Z-Shave. Exploiting Z-Wave downgrade attacks,” PenTestPartners, 23 May 2018. [Online]. Available: https://www.pentestpartners.com/security-blog/z-shave-exploiting-z-wave-downgrade-attacks/. [Accessed 4 May 2022].

[53] Shodan, “Search result of Home Assistant,” Shodan, [Online]. Available: https://www.shodan.io/search?query=Home+Assistant. [Accessed 4 May 2022].

[54] silvrr, “Securing Your Home Assistant Instance,” Home Assistant, September 2021. [Online]. Available: https://community.home-assistant.io/t/securing-your-home-assistant-instance/336908. [Accessed 2 May 2022].

[55] Shodan, “Search results of openHAB,” Shodan, [Online]. Available: https://www.shodan.io/search?query=openHAB. [Accessed 4 May 2022].

[56] T. Goodreau, “7 Actionable Tips to Secure Your Smart Home and IoT Devices,” IEEE COMPUTER SOCIETY, [Online]. Available: https://www.computer.org/publications/tech-news/trends/7-actionable-tips-to-secure-your-smart-home-and-iot-devices. [Accessed 28 April 2022].

Appendix 1 – Non-exclusive licence for reproduction and publication of a graduation thesis[1]

I Siim Soopalu

1. Grant Tallinn University of Technology free licence (non-exclusive licence) for my thesis" Smart home system framework selection", supervised by Andres Rähni.
   1.1. to be reproduced for the purposes of preservation and electronic publication of the graduation thesis, incl. to be entered in the digital collection of the library of Tallinn University of Technology until expiry of the term of copyright;
   1.2. to be published via the web of Tallinn University of Technology, incl. to be entered in the digital collection of the library of Tallinn University of Technology until expiry of the term of copyright.
2. I am aware that the author also retains the rights specified in clause 1 of the non-exclusive licence.
3. I confirm that granting the non-exclusive licence does not infringe other persons' intellectual property rights, the rights arising from the Personal Data Protection Act or rights arising from other legislation.

09.05.2022

1 The non-exclusive licence is not valid during the validity of access restriction indicated in the student's application for restriction on access to the graduation thesis that has been signed by the school's dean, except in case of the university's right to reproduce the thesis for preservation purposes only. If a graduation thesis is based on the joint creative activity of two or more persons and the co-author(s) has/have not granted, by the set deadline, the student defending his/her graduation thesis consent to reproduce and publish the graduation thesis in compliance with clauses 1.1 and 1.2 of the non-exclusive licence, the non-exclusive license shall not be valid for the period.