Contents

Master Thesis

Abstract

The presence of connected devices in our environment is increasing. These devices form a network often called Internet of Things (or IoT for short), where everything from lightbulbs to thermostats can be controlled by an app or by another device. These services make a lot of that data available to the end user but also to malicious parties due to the devices leaking more data than

intended or by bad design. This puts the end user at risk, violating its privacy and leaking sensitive data. One simple and obvious way to prevent leakages and misuses of personal data is to collect less of this data, a principle known as data minimisation. However, this solution is rarely used in practice because of business models relying on personal data harvest on one hand and because of the difficulty to enforce it once it is defined what is actually needed to provide a service.

Acknowledgements

Introduction

Motivation

Aim

In this thesis I investigated ways to improve privacy in a special kind of IoT devices known as Wireless Sensor Networks (WSN). WSN are networks of autonomous sensors and actuators. The goal to enhance privacy for this kind of devices will be addressed by relying on data minimisation. This means the project sought to improve privacy in distributed networks by limiting the amount of personal data being processed.

Scope and Limitations

Thesis Structure

Background

Wireless Sensor Network (WSN)

Over-collection

Over-collection is the event when a process collects more data than it requires to function properly. The event is very example-specific and will depend heavily upon collection-process but some basic examples of over-collection can be:

Data Minimization

As defined by the EDPS (European Data Protection Supervisor); "The principle of "data minimization" means that a data controller should limit the collection of personal information to what is directly relevant and necessary to accomplish a specified purpose. They should also retain the data only for as long as is necessary to fulfil that purpose. In other words, data controllers should collect only the personal data they really need, and should keep it only for as long as they need it."

Promela

Promela stands for **PR**ocess **ME**ta **LA**nguage. It was used for the development of this project since it allows for formal verification and I had previous experience in it.

SPIN

SPIN stands for **S**imple **P**romela **IN**terpreter. It's the verification program that was used for this course since it's explicitly built for verifying promela code.

Related Work

Smart City

Specification

Decisions

A decision is a control message sent through the network forcing an action to be taken, in this report that's mainly when over-collection is occuring. This can for example mean that the server is telling one (or several) collection nodes that they should shut down or wait until sending again. If the nodes are also computing data they can still be kept doing so but without sending communication throughout the network until notified again. The source of the decision depends on if the network has a centralized processing unit (f.e. a master-slave relation) or if computations are decentralized and each processing unit makes decision on their own and uses communication to forward processed data to a storage server.

Defining the models

Formal Development

As suggested when developing formally, I started out with a simple model as possible. So the model expects that each actor acts as intended, without malicious users or failing proceedures, and that all messages gets through, e.g. no lossy channels.

The basic features of the system is a set of collection nodes, a server and an environment.

Image 1: *As seen in the image the messages that continues the flow are noted 'ack'. The picture describes a centralized system.*

The environment is a simple procedure, it starts off by generating random data and servering it to a requesting node. This is a modelling simplification, since the node should rather extract the data from the environment rather then be waiting for a response. But this was made to simplify the system and made the environment easier to manage as a shared resource for the nodes.

The collection nodes follows the same proceedure, where they start by 'requesting' data from the environment. Then the model was split into two different choices depending on the structure of the system. If the system was decentralized, the decision could be taken by a node. Meaning that then the node would first check if the data requested from the environment was below a certain threshold. Then it would send the data along to the server with either a message 'go', meaning that all was fine, or a message 'stop' meaning that over-collection was occuring and it was going to stop collecting and the other nodes should do the same.

If the system was a centralized one, the node would simply pass the data forward to the server. Then it would wait for a response to see if it should continue collecting or if it should stop. This describes a typical scenario for the system, so from this a behaviour model for each actor can be defined.

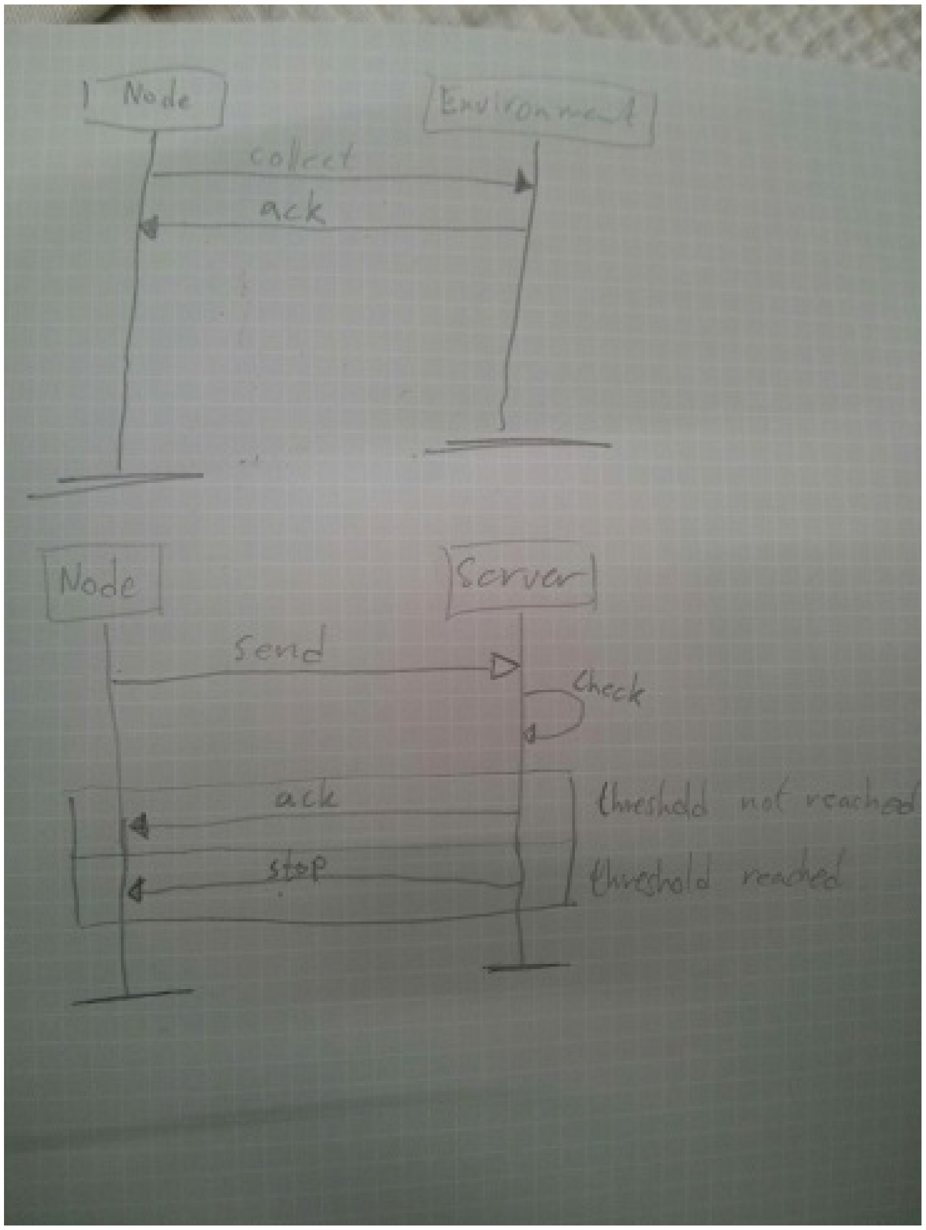
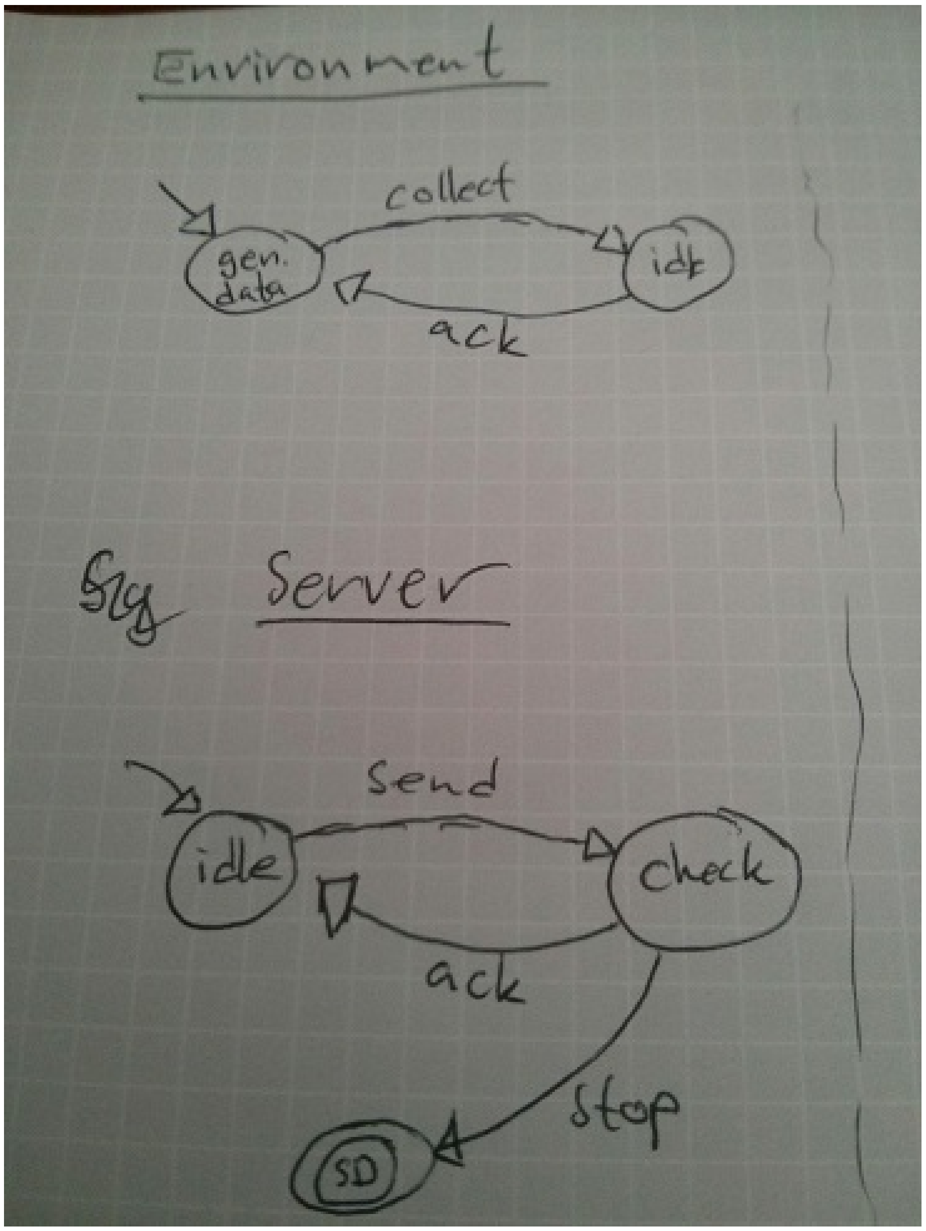


Figure 1: Img_1


Environment
collect
gen. data
idle
ack
Server
send
idle
check
ack
stop
SD

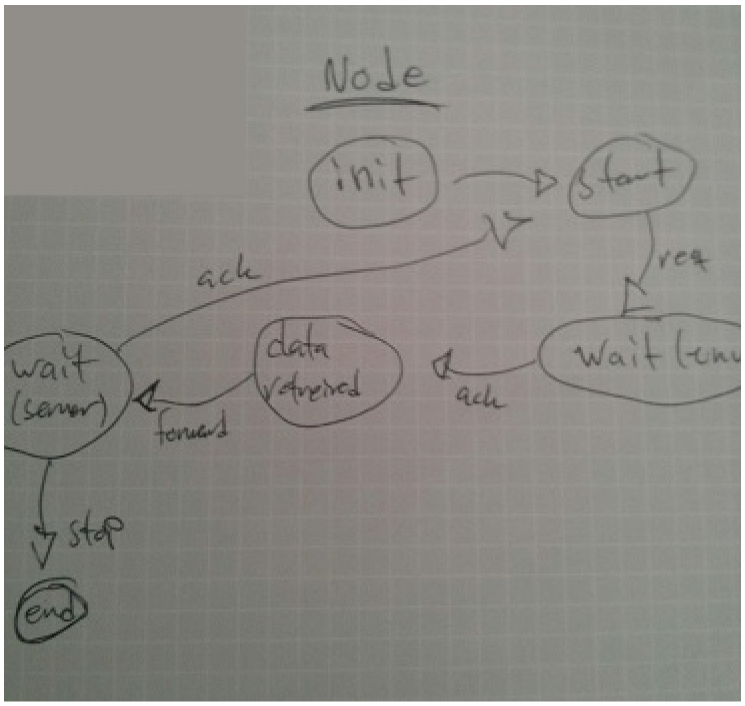


Image 2 & 3: *The behaviour model of the environment and the server is to the left and the one for the nodes are to the right.*

The behaviour model of the nodes can explain the behaviour of the entire centralized system, since the node 'communicates' with all the involved parts and is the one taking (being effected of) all the actions. What distinguises the decentralized system and the centralized can be noticed in the behaviour model for the nodes, in the state marked 'data retrieved'. Here the node would check the data on it's own and notify the server of the action being taken, but the behaviour model still works for both cases.

explain what this model doesn't consider and possible variations for the decisions.

LTL Properties

explain how the LTL properties were defined

Initial Model

explain the scope of the initial model

The first model had properties for **correctness** and **liveness**. Due to the simplicity of the model, made both of them also rather simple to manage. The correctness property was stated as follows:

When over-collection has occured (the decision is taken), the system should stop collecting.

Over-collection and the decision were rather interleaved in this system, the decision was taken at the same time the data was checked, therefore the brackets. The liveness property was stated as:

The system should collect until over-collection has occured.

Extended Model

Design

Algorithm design

Decisions

Implementation

Code Generation

Analysis

Verification

LTL Properties

Satisfaction

Discussion

discuss the thesis work

Conclusion

conclude the results

Ethics

discuss the/some ethics involved

Table of References