Implementation of a Security System based on RFID and WSN technology

Nils Timotheus Kannengießer **Helmut Dispert**

Kiel University of Applied Sciences, Germany
Faculty of Computer Science and Electrical Engineering
Nils.T.Kannengiesser@student.fh-kiel.de, Helmut.Dispert@fh-kiel.de

1. ABSTRACT

Over the last few year we have observed a convergence of wireless sensor networks (WSNs) and RFID systems. In the next generation "Super RFID technology" traditional passive tags will be replaced using dedicated WSNs that extend standard tags to allow data acquisition and data storage, e.g. for monitoring purposes.
In this paper we suggest a new application for Super RFIDs in security systems with an emphasis on theft control and prevention.

A standard Sun Microsystems SunSpot WSN system is used to detect the illicit removal of objects. The WSN nodes are equipped with a three-axis linear accelerometer capable of precisely measuring motion. A master WSN-node in combination with an RFID reader allows to control and supervise a group of slave-nodes that are attached to the objects under surveillance.
We will introduce the SunSpot technology, demonstrate the effectiveness of the complete system and discuss future application scenarios.

2. INTRODUCTION

Over the last few years we have observed a convergence of wireless sensor networks (WSNs) and RFID systems. In the next generation "Super RFID technology" traditional passive tags will be replaced using dedicated WSNs that extend standard tags to allow data acquisition and data storage, e.g. for monitoring purposes.
In this paper we suggest a new application for Super RFIDs in security systems with an emphasis on theft control and prevention.

A standard Sun Microsystems Sun SPOT WSN system is used to detect the illicit removal of objects. The WSN nodes are equipped with a three-axis linear accelerometer capable of precisely measuring motion. A master WSN-node in combination with a RFID reader allows to control and supervise a group of slave-nodes that are attached to the objects under surveillance.

3. TECHNICAL BASICS

Beside Sun SPOTs as the controller unit a RFID module had to be chosen. We decided us for the M1 by Skyetek, because of the low current consumption and various supports of RFID standards. The M1 uses 13.56MHz for transmission and supports e.g. Mifare Ultralight or Standard.

For Sun SPOTs there are various add-on boards [1] available for self-building, like the eDaq, eFlash, eProto, eProtomega, eSerial or the eUSB-Host. For all these add-on boards the sources are made available by Sun Microsystems for free (open source).

Specifications of the Sun SPOT [2]:

Processor	180 MHz ATMEL AT91RM9200 (3.0V, 44mW)
RAM	512KB
Memory	4 MB NOR Flashmemory
Interfaces	SPI, USB, I2C, DMA, radio (2.4GHz)
Future interfaces*	USART, Ethernet MAC, MMC, I2S, USB-Host, TWI
Addon Board	Standard: eDemoboard
4 power modes	run (70-120mA), shallow sleep (24mA), deep sleep (33µA), special lower power mode(6,4mA)
Battery	Li-Ion 720mA/h
OS	Squawk VM

Table 1 - Specifications Sun SPOT

* Currently not accessible, but physically pointed out to the add-on interface

Normally the eDemoboard is attached to the Sun SPOT. It consists of a few devices like temperature sensor, accelerometer, light sensor, LEDs, switches and of cause the interface-pins to mainboard.

D0 und D1 of the normal IO-pins (eDemo-board) are used for serial connections. There are also high current pins (H0-H3) and it's possible to measure analogue voltages by using the pins A0-A3.

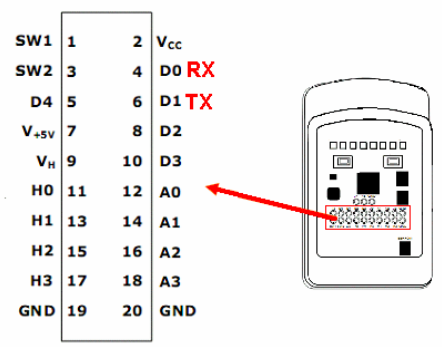

Figure 1 – eDemoboard interface [2]

Sun SPOTs use the Squawk VM as kind of OS. The java applications run directly "on-the-metal". There are a few differences between Standard Java VM and the Squawk VM (Java ME). The main difference is that the Squawk VM is mainly written in Java.

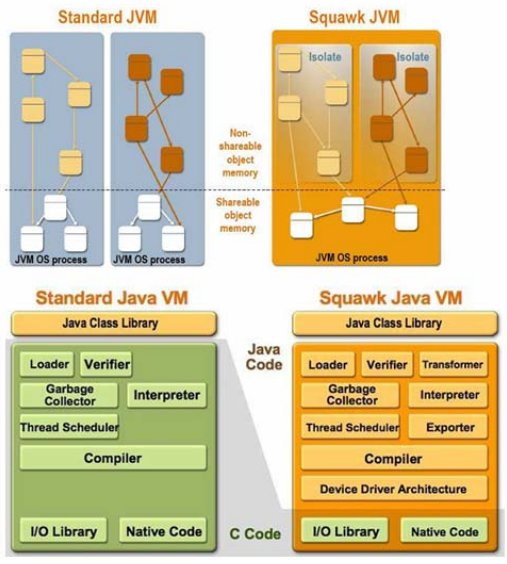

Figure 2 – Squawk VM vs. Java SE [3]

The networking of the Sun SPOTs is based on 802.15.4 protocol with lowpan and uses the 2,4 GHz band. For addressing 64bit IEEE addresses are used. At the moment only the last 16-bit are changed (0014:4F01:0000:XXXX). Sun SPOTs also use mesh routing with a standard hop count of 15, which is changeable so that it's possible to address nearly unlimited devices. Because of using channel 26 (2.480GHz) there might be problems
with Bluetooth adapters. Sun SPOTS should support IPV6 in the future, but currently there is no implementation.

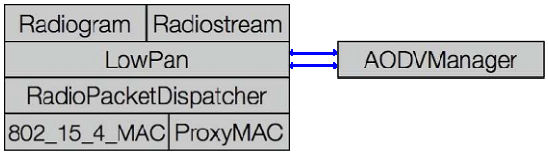

Figure 3 – current network layer [4]

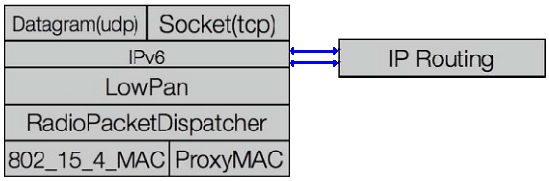

Figure 4 – network layer (in the future) [4]

Sun delivers some software with the Sun SPOTs like the Netbeans IDE for development. Sun SPOTs are administrated by using the "Sun SPOT Manager" and can also be emulated in the Sun SPOT Emulator called "Solarium".

4. SYSTEM REALISATION

The whole system should consist of two Sun SPOT types. One called "master-spot" and another group called "client-spots". The system itself is made of a Sun SPOT and the M1 RFID module with a speaker unit (master) or only a speaker unit (client).

The first RFID card which is used with the system will be the RFID card of the owner.

While the master-spot is used to start the monitoring, the client-spots only wait for requests. The master-spot is polling the clients every 5.5 seconds. In the other time both systems will try to save much current as possible by using the low power modes.

All spots will look for moving, while the master-spot is polling every client. If a moving is detected the moved spot will alarm the

master, who will itself enable the alarm on all clients. In case of an alarm the master spot will try to send a http request e.g. a message to a cellphone. The alarm can be turned off by using the RFID card of the owner again.

Of cause it's possible to disable the monitoring by pointing the RFID card of the owner to the reader field.

Current consumption:

In the current implementation the systems are turned on about 15 minutes and sleep about 45 minutes per hour. The master-spot will consume up to 232mA and the client up to 132mA. In this configuration the system will stay online for about nine hours.

5. SUN SPOT AND M1 PROBLEMS

Sun SPOT: Encryption

Encryption is not useable at the moment, since the current technology preview will produce a hidden "bad alert level" error in complex applications (the spot seems to crash). In very small applications it was possible to use the encryption. The encryption is based on ECC (Elliptic Curve Cryptography).

Sun SPOT: Power mode "deep sleep"

As the Sun SPOTs choose the power modes itself, the programmer can only try to set the conditions for e.g. deep sleep. In this project it was not possible to get the Sun SPOT into "deep sleep"-mode, because of a running, unidentified system thread. The coders confirm that there might be problems with this mode [5].

Sun SPOT: No timeout for WriteUTF()

A radio connection is established by using the open function on each Sun SPOT side. If one of the spots is not reachable there may occur a "NoRouteException". If not, the spot believes it's possible to send the information to the partner spot. In this application a client might be stolen and a problem was measured in the WriteUTF function that the master-spot will hang sometimes if a client is illegally removed.

M1: Wrong detection of RFID cards

In random tests it was determined that the M1 detected wrong IDs, if using the same Mifare Ultralight (ID: 0455FC61EE0280). This error is reproducible in an amount of tests.

CMD	TYPE	ID	CRC
14	04	88**0455FC**	4477
14	0A	**0455FC**61EE0280	1EF9

Table 2 - Detected IDs by M1 module (same card)

6. CONCLUSION

Sunspots are still experimental. With a period of vocational adjustment it's possible to program the spots easier than ASM/C devices, but Sun SPOTs have grave problems like too high current consumption, some faulty or incomplete functions (encryption) and a lot of missing opportunities like e.g. IPV6 support.

The attached documentation is still incomplete, but the very well support of developers in their forum is to underline.

The M1 by Skyetek also seems to have some technical issues (wrong detection).

7. REFERENCES

[1] https://spots-hardware.dev.java.net
[2] Sun SPOT documentation, SunSPOT TheoryOfOperation.pdf
[3] [Sun Microsystems, https://www.enterpriselab.ch/cms /images/stories/presos/preso2.pdf
[4] Sun Microsystems, https://spots-network-library.dev.java.net/stack-overview.html
[5] Developerforum, https://www.sunspotworld.com /forums /viewtopic.php?t=995

Further References:
Nils Kannengießer, Bachelor's Thesis, 2008