Journal of Universal Computer Science, vol. 15, no. 5 (2009), 992-1006
submitted: 1/8/08, accepted: 28/2/09, appeared: 1/3/09 © *J.UCS*

Agent Platform for Wireless Sensor Network with Support for Cryptographic Protocols

Peter Pecho, Frantisek Zboril jr.
Martin Drahansky, Petr Hanacek
(Brno University of Technology, Faculty of Information Technology
Brno, Czech Republic
{pecho, zborilf, drahan, hanacek}@fit.vutbr.cz)

Abstract: This paper deals with a description of wireless sensor networks at the beginning. Further follows the introduction to the agent platform suitable for wireless networks. Mobile code and sensor networks suffer from considerable security problems. Our proposal of countermeasure is based on combination of smartcards with sensor nodes. Smartcards as a tamper resistant devices offer solution for the most of commonly required security objectives. Analysis of proposed hardware cryptographic platform includes link level communication, transport protocol description, application interface description and demands for power consumption. Today smartcards are highly standardized devices that offer common communication interface and platform could be used with various smartcards in accordance with ISO/EMV standards. At the end, we discuss the combination of agents in wireless sensor networks in conjunction with usage of cryptographic protocols for securing of wireless networks.

Key Words: agent, wireless sensor network, smartcard, cryptography, protocol
Category: SD C.2.0

1 Introduction

Wireless sensor networks (WSN) belong to data-centric networks, their importance lie especially in data collection, transport and evaluation (Fig. 1). For the programming of such networks, there are used tools based on TinyOS(2)/nesC [Lev 2006] environment at the moment. The programmer writes a program that is a set of components which react on some events raised from sensors, transceivers or other hardware elements, or react to some commands send by another program component. This event-based approach is quite appropriate for creation of WSN application due to resources limitation of the particular WSN elements, concretely energy limitation due to relatively short battery life.

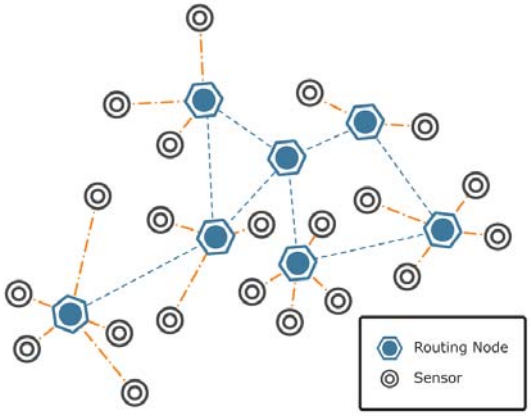

Figure 1: *Example of a wireless sensor network [Pur 2008]*.

Sensor node, usually called "mote", consists typically of a sensor board, a radio module and a control board containing a microcontroller (Fig. 2). To introduce such platform we describe the one that we and many other use for WSN development. It is a solution produced by CrossBow[1] company and they called their devices to be MICAz. Their motes consist of radio board working at the physical level in accordance with ZigBee specifications [ZigBee 2008]. The radio board also contains an ATMEL microcontroller, today they use ATMEL128L and ATMEL1281 microcontrollers. The radio board can be connected to one of their sensor board.

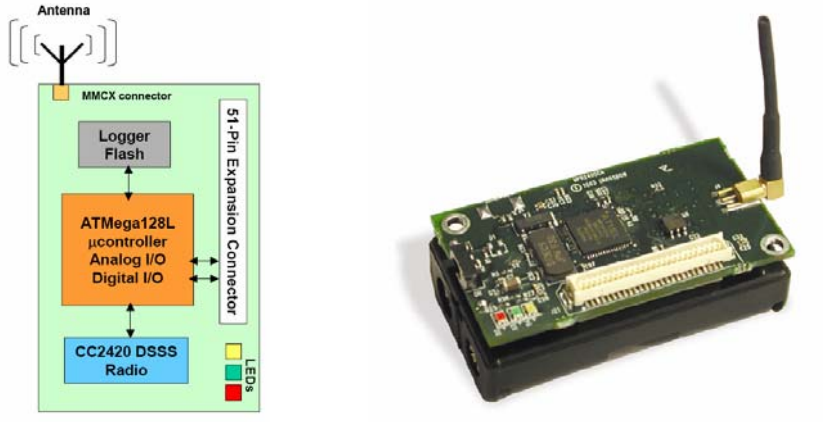

Figure 2: Architecture and the device of CrossBow mote.

Contemporary microcontroller based systems mean quite low level tools for their programming. Applications are close to hardware and their kind of work is mostly

[1] http://www.xbow.com

data gathering from the sensors, processing them and transferring through the wireless net. However, such approach can be done at the assembly language level, or there exists a tool, which brings more conformability for WSN application development.

Such tool is a combination of nesC language with TinyOS system. In fact, the nesC is a dialect of the C language developed in that form, which is appropriate to this area. Program written in the nesC is a set of components that are linked together at the compilation time. All the components can use other components and then the application is built as a structure of such components. Each component is either a configuration or module. The configurations address other components and wire them together. Modules are then implementations of particular interfaces. The implementation of modules consists of commands, which are sent from a component downwards toward the motes hardware and on the other hand there may raise events generated by the hardware (sensor, transceiver, serial port etc.), which are propagated upwards on the tree structure and it can be processed by a module event handler.

Although the name "TinyOS" may be understand so that it is another operating system, but in fact it is a library of components for many of mote's platform that can be used by a programmer. Today there is second version of the TinyOS system being developed. Usually the TinyOS provides interfaces for implementation of modules.

The connection between WSN and the generation and use of keys is intended for enhancement of the security in wireless sensor networks [Per 2004] [Wal 2006] by taking of smart cards, which are used in the secure applications (are proven for secure solutions) and are very cheap. However, this approach could be combined with another sophisticated method – agent systems. If we combine such subparts – i.e. WSN with security cards and with agents – we get a new approach in the field of securing of wireless sensor networks, using very cheap solution and having some methods (agents), which could be used e.g. for the detection of a threat etc. The purpose of this article is to describe such solution and to show the possibilities of such combination.

In this text we describe how to make applications for the MICAz devices which uses agent based principles together with some cryptographic protocols. We demonstrate how the agent platform is built first in the chapter 2. Then we identify possible weak spots in the usage of the WSN, we show possible threats to the applications using this kind of systems and we also present some solutions based on cryptography. That is presented in the chapter 3 together with some discussion of usage of the smart cards as the computational devices for ciphering and deciphering. Chapter 4 shows some applications where the mentioned principles can be used and the chapter 5 brings some concluding remarks.

2 Artificial Agents and Agent Platforms

In general, artificial agents are advanced technologies in system development and engineering. The idea of such agent is closely studied for more than twenty years. The state of the art in the agent technologies includes some well developed reasoning principle based on agent's mental states, algorithms for coalition forming as well as several implementation tools. Today agent architectures vary from simple rule-based systems, automata and their layered combinations to modern systems based on agent mental states. Shoham's Agent-0 and BDI systems based on Bratman's philosophy of

intentions [Bra 1987] and developed as computable system has been originally introduced by Rao and Georgeff [Rao 1996, Rao 1995].

2.1 Agent Applications

However many theoretical studies work with the term "agent" and consider such agent to be a part of networking and reasoning process. Agents can be found in the methods for dynamic topology discovery [Nas 2007], data integration [QiH 201] or tracking [Che 2002]. Agent abilities include here data collection, storage, manipulation and autonomous reasoning. Mobility and flexible reasoning in unknown or partially known environment are those properties of agent systems which are very useful for usage in the WSN systems.

For the realization of agent ideas in the network of wireless sensor devices, we need to identify which role they should play there. The architecture will be selected on the basis of this appropriate agent. Advantages of the agents have been mentioned already. It is also fair to say something about disadvantages of this approach. The problem is that there exist some limitations in the area of wireless sensor networks, what significantly reduces usage possibility of current mostly Java-based agent technologies. We summarized these problems in the following points [ZZ 2008]:

- Sensor motes are based on microcontroller with relatively small programmable and dynamic memory.
- WSN are expected to work for relatively long period of time without battery exchange. For this reason the motes spend major time in sleep-mode. There probably should not be common robot/agent infinite loop that manages agent behavior on the very top level.
- Typical agent that works in a task for WSN is mobile. It means that model and consequent realization of such agent should be developed as a mobile object.
- WSN are ad-hoc networks that need to deal with net structure changes. Agent platform should provide information about available communication channels and migration possibilities.

2.2 Agent Implementation

Due to the significantly small dynamic memory – we use motes working with AtmeL128L which has just 4kB dynamic memory – the agent program have to be short as possible to avoid wasting of memory, but be expressive enough to enable agent's practical reasoning process at a sufficient level. In this moment there exists an agent framework for WSN like Agilla [Lia 2005] that allows implementation of mobile agent in *assembly language* manner. However, we have developed another one for the reason of making the framework, which is closer to above mentioned agent development methods. Our agent is a code containing some data and a control program interpretable on the agent platform. The code is written in Agent Low Level Language (ALLL), which we developed and we hope it will be appropriate for such kind of applications.

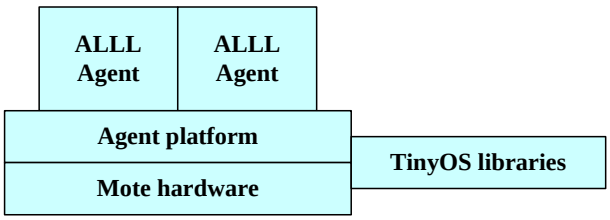

Figure 3: Hierarchy of platforms and agents.

Each agent resides at one moment at one platform. Agent platform is located on a hardware device or more distributed devices and serves as a background for particular agents. The most contemporary platforms respect standard released by Foundation for Physical Intelligent Agents (FIPA) [FIP 2008] that states, which modules the agent platform should implement. Our approach uses some of these modules, but not all from this specification, because our agent is built at a lower level of abstraction in comparison to those considered in the FIPA standard. Anyway each platform resides at one mote and consists of the following modules:

- *Agent management unit* – is responsible for agent(s) existence(s) at the platform. It also controls agent transports among the platforms.
- *Platform services* – services provided to the agents. It includes computation (a set of functions), interpreter control like waking up agents, when an input is received, and it also includes some platform variables that are accessible to particular agents.
- *ALLL interpreter* – interpreter of the ALLL language that manipulates with agent program and possibly interacts with other modules.
- *Input/output sensor interface*.
- *Message transport system* – responsible for correct agent and data transmission.
- Hierarchy of the system is shown in Figure 3. Hardware device works with code stored in its flash memory. The code is implementation of agent platform written in nesC for TinyOS v.2. Particular agents are codes written in APL containing agents plan, registers, input buffer, knowledge base and plan base and they are stored in mote's dynamic SDRAM memory.

3 Security of Wireless Sensor Networks

Beside the agent, platform design and implementation, we address issue of communication that comes out with a security issue. All agent platforms have to realize the most important requirements for wireless sensor network protocols: low communication and computational cost, scalability and data integrity. Depending on the application and the usage of WSN – motes are mostly scattered in a hostile environment – there could be also the additional requirement for security protocols: authentication (information is genuine and comes from declared source), data integrity (information can not be modified without authorization), confidentiality

(information can not be read by an unauthorized subject), availability (information must be available when it is needed) and for some applications also non-repudiation (one party of transaction can not deny having received a transaction nor can the other party deny having send the transaction) [Per 2004].

Conventional architectures, such as TinySec, are compromise between information security and limited resource requirements. Strong cryptography and algorithms used on the Internet are usually too computational consuming, so they had to be replaced by alternative solutions, mostly by symmetric ciphers and stream ciphers. These solutions could guarantee authentication, confidentiality and integrity at the expense of shared global cryptographic key. But usage of one shared key could be a problem, if the motes (sensor node) are placed in a hostile environment. Typical sensor motes are not tamper-resistant due to the cost factor. Also a key distribution schemes based on public key cryptography or elliptic curve cryptography are not suitable, due to their computational cost.

3.1 Threats

Computing in sensor networks pose problems to networking because of changing physical location that requires continuous reconfiguration of the data links. If the connectivity cannot be always maintained, applications also have to handle extended off-line periods as well. The code in sensor network nodes generally runs in the hostile environment. Running code in hostile environment means that programs or program fragments are executed by computers that could have different interests than an author of the code. Such programs are also called mobile code.
Mobile code however, suffers from considerable security problems. From the security point of view there are a number of threats for the sensor network. The main threats include [Han 2005]:

- *Eavesdropping of communication* – the attacker can obtain all available information about the sensor network, including security relevant information and position of individual nodes from the eavesdropped messages.
- *Replay attack* – the attacker is not able to break the cryptographic protection of messages but is able to re-send previously sent genuine messages.
- *Jamming the communication* – injecting noise in the radio channel can be used to make the communication within the network impossible. This attack could be made more difficult by using spread spectrum radio channels.
- *Denial of service* – the attacker nodes can produce a large number of demands to fill-up network nodes or exhaust the node batteries.
- *Node masquerade* – the attacker produces its own network nodes and attempts to connect these bogus nodes to the existing network.
- *Falsification of sensor data* – the attacker nodes can produce a counterfeit sensor data that they send to other sensor network nodes for distortion of sensor data.
- *Tampering with the node computer* – the attacker is able to use a direct logical or physical manipulation with the node to change the behavior of the node and, consequently perform further attacks to the rest of the network

nodes. The attacker can also acquire the cryptographic keys stored in the node computer.

The security countermeasures fall into three areas – protection of radio channel, protection of messages, and protection of node hardware. Making the motes more tamper resistant is usually not economically viable. Therefore we must sometimes admit that the attacker could compromise the sensor nodes.

3.2 Role of Cryptography

Cryptography has important role in the design of wireless sensor networks that usually work in a hostile environment with the different threats. The application of cryptographic mechanisms can help achieve security objectives such as confidentiality, data integrity, authentication, and non-repudiation. The cryptographic mechanisms used in sensor networks systems include [Han 2005]:

- secret key encryption/decryption
- one-way hash functions
- challenge-response cryptographic protocols
- digital signatures.

Secret key (symmetric) encryption methods provide confidentiality, authentication and data integrity. Asymmetric algorithms could provide all these objectives including non-repudiation but at the expense of much higher time and resource consumption. Therefore, symmetric algorithms are generally preferred. Data integrity and authentication are achieved by applying well-known hash and MAC algorithms, such as CBC-MAC and SHA.

Research highlights that public key cryptography, especially RSA algorithm, is not well suitable for wireless sensor networks. Much more suitable is the usage of ECC algorithm and 160 bit keys, that is as secure as RSA-1048 but at a lower power consumption. While the application of the stronger ECC-224 still seems to be feasible, the time and power consumption for the equivalent RSA-2048 is far beyond the acceptable level [Amin 2008].

3.3 Role of Tamper-Resistant Hardware

Most widely used kind of trusted device is smart card. Smart cards offer very cheap implementation of the one of the security concepts, this concept is called “tamper resistant hardware”. This term can be described like a hardware resistant against physical attack or as secure hardware. Secure hardware is hardware module ([And 1972], [Tru 1985]) equipped with microprocessor, which contains some secured data and algorithms, which manipulate with these data. These features can be used in two main ways:

- Secure hardware contains data, with which can be manipulated by the specific way. As example we can use telephone smart card, which contains impulse counter, which can be only decreased.

- Secure hardware contains secret cryptographic key, which can be used only for encrypting data and this key remains in the card and can't be read. As an example we can use authentication smart card, this card authenticates itself thanks to encrypting data with his secret key.

This secure hardware can't be cloned or emulated, because these features can't be replaced just by software without special hardware. For example: if we will simulate the telephone smart card, we can't ensure that someone wouldn't rewrite counter number in our program and as well we cannot ensure that in our software implementation of smart card someone wouldn't rewrite our secret key.

Tamper resistant devices are evaluated by means of FIPS PUB 140-1 [FIPS 140-1] or FIPS PUB 140-2 [FIPS 140-2], U.S. government computer security standards. FIPS 140-2 defines four levels of security – "Level 1" to "Level 4".

The lowest security is provided by Level 1, example of such device is a personal computer (PC). Security Level 2 enhances physical security mechanisms of Level 1 device by adding tamper-evidence properties. Level 3 security has to have strong enclosures and tamper a detection/response circuit that zeroizes all plaintext sensitive data when the removable covers/doors of the cryptographic module are opened. The highest level of security is provides Level 4, such devices are secured against compromise due to environmental conditions outside of the device's normal operating ranges of voltage and temperature.

Today smartcards offer security properties consistent with Level 2-3 of FIPS 140-2.

3.4 Possible Solutions

Agent platforms are usually based on a conventional hardware with no hardware or software cryptographic support. Cryptographic abilities of such platforms could be improved in several ways:

- Cryptography algorithms could be implemented into the firmware of sensor node. This least expensive solution provides usually only the private key cryptography at low speed. Performance of this cryptography could be sufficient for simple applications. Security of stored cryptographic keys is usually very low.
- Sensor nodes could be extended with an auxiliary microprocessor(s) implemented parallel computing of the cryptographic algorithms. Performance and security of this solution depends on auxiliary microprocessor(s).
- Sensor nodes could be improved by an add-on secure microcontroller with secure memory and hardware crypto engine supporting AES and 3DES encryption standards. The encryption engine increase encrypted communication speed up to several Mbps. Regrettably, current crypto-accelerators suitable for sensor networks have no support for public key cryptography.
- Sensor node could be extended with an FPGA module implementing fast symmetric and asymmetric cryptography algorithms. This solution offers

high speed encryption and key generation at a highest price and power consumption (depends on FPGA module).

- Smart cards (Fig. 4), as the tamper resistant devices, could be used as crypto accelerators and also as a secure storage for cryptographic keys. These crypto-accelerators ordinarily support symmetric and asymmetric cryptography together with low power consumption. Smart card consumption can be also decreased powering them up only for cryptographic operations. Furthermore, cost of high-end smart cards is lower than cost of the same feature crypto-processors and FPGAs.

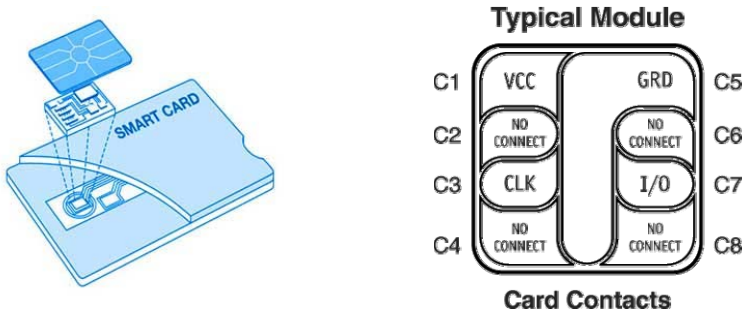

Figure 4: *Smart-card structure [Tir 2008]; Typical smart-card contacts [Sma 2008].*

3.5 Smart Cards

As the smart card family consists of two broad categories – memory cards and microprocessor cards – we will be interested only in the contact microprocessor cards (according ISO 7816 standard [7816-1, 7816-3, 7816-4]) as they are freely programmable. The functionality of microprocessor cards is restricted only by the available storage space and the performance of the processor.

The only communications between a smart card and the outside world take place via a bidirectional serial interface. Originally, solutions for data transmission and reception via this interface were controlled exclusively by software, without any hardware support. Current solutions use UART (universal asynchronous receiver-transmitter) component independent of the processor. Default transmission rate of smart cards is 9,600 kbit/s, but most of the cards can transmit up to rate 115 kbit/s with a 3.5 MHz clock.

Smart cards communicate can communicate by means of byte-oriented T=0 protocol or by block-oriented T=1 protocol. Other transmission protocols are not standardized and therefore they are not usually supported. There protocols transmit APDU (application protocol data unit) blocks. APDUs can be classified into (terminal → smart card) *command APDUs* (Fig. 5) and (smart card → terminal) *response APDUs* (Fig. 6) [Rankl 2003].

CLA	INS	P1	P2	Lc	Data	Le
← Mandatory header →				← Optional body →		

Figure 5: Command APDU structure [EMV2000].

Data field	SW1	SW2
← Optional body →	← Mandatory trailer→	

Figure 6: Response APDU structure [EMV2000].

Command APDU consists of instruction class (CLA), instruction code (INS) and two parameters (P1, P2) in the mandatory header. Optional body carries command data (Lc, Data) and maximum number of data expected in data field of response (Le). Response APDU carries optional data and two mandatory status bytes (SW1, SW2).

Examples of commands of Gemplus GPK smart cards [GPK 2001]:

```
// Select application ("CRYPTO1")
00 A4 04 00 07 'C' 'R' 'Y' 'P' 'T' 'O' '1' FF // PC -> Smartcard
90 00  // OK                                  // PC <- Smartcard

// Verify PIN (1111)
00 20 00 01 08 01 01 01 01 FF FF FF FF FF     // PC -> Smartcard
90 00  // OK                                  // PC <- Smartcard

// SelectCryptoContext (7)
80 A6 07 32                                   // PC -> Smartcard
90 00 // OK                                   // PC <- Smartcard
// PutCryptoData ("<data>")
80 DA 01 05 07 55 05 <data>                   // PC -> Smartcard
90 00 // OK                                   // PC -> Smartcard

// PK_Sign
80 85 00 00 80                                // PC -> Smartcard
<response> 90 00 // Response & OK             // PC <- Smartcard
```

Power consumption of smartcard is similar (or lower) to consumption of wireless sensor nodes – see fig. 9 and fig. 10. Active supply current of ATmega128 could reach 10 mA. MICAz sensor nodes, we are using, consume 5 mA in an active state [ATM].

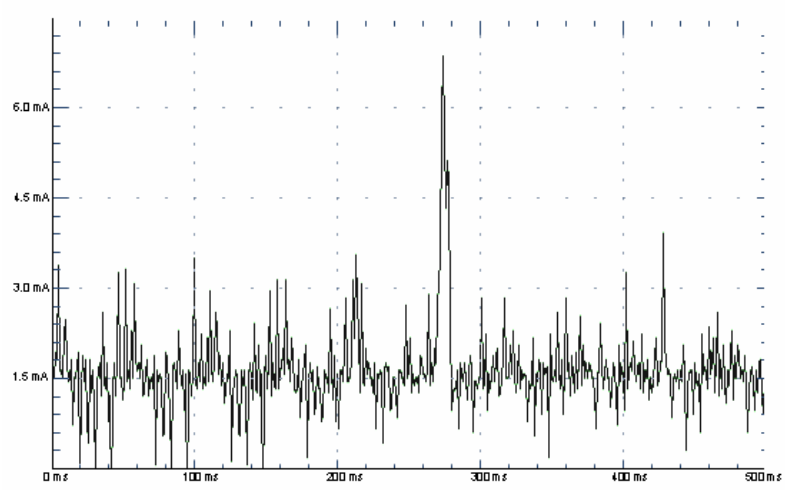

Figure 9: Supply current of 3DES encryption using GemXpresso R4 smartcard.

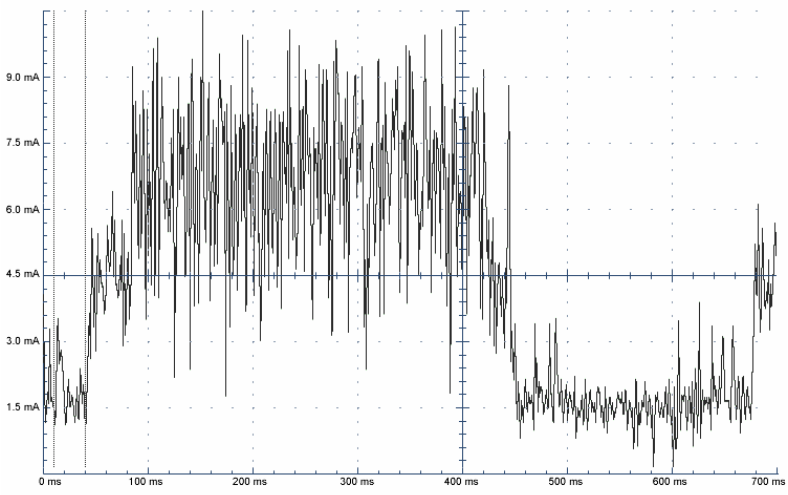

Figure 10: Supply current of RSA-1024 signature using GemXpresso R4 smartcard

3.6 Smart Card Interface and TinyOS

Firmware of sensor nodes is typically implemented using specialized WSNs operations system TinyOS and nesC language. Sensor node logic is based on events raised from sensors, transceivers or other hardware elements.

Smart card is connected to the sensor node via UART (serial interface) and one auxiliary pin on the data bus for sending of a reset signal. Before using of connected smart card, it's necessary to send reset signal. If the smart card gets through this reset successfully, answer-to-reset (ATR) block is send to the sensor node. ATR describes

supported transmission protocol, transmission rate, timing and other serial interface settings.

Described solutions was successfully tested on MICAz platform (ATMega128 processor) and Gemplus GemXpresso Pro R3 (E64 PK) smart card.

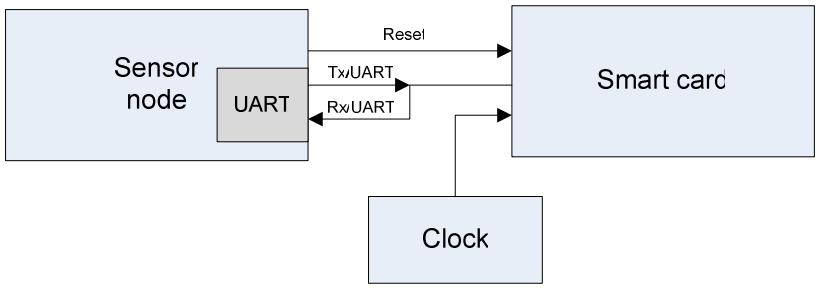

Figure 7: Connection scheme of sensor node and smart card.

4 Resulting Architecture and Application

The above described parts could be combined into one wireless sensor network, using agents and smart cards as cryptographic processors for data transfer encryption – see Fig. 8.

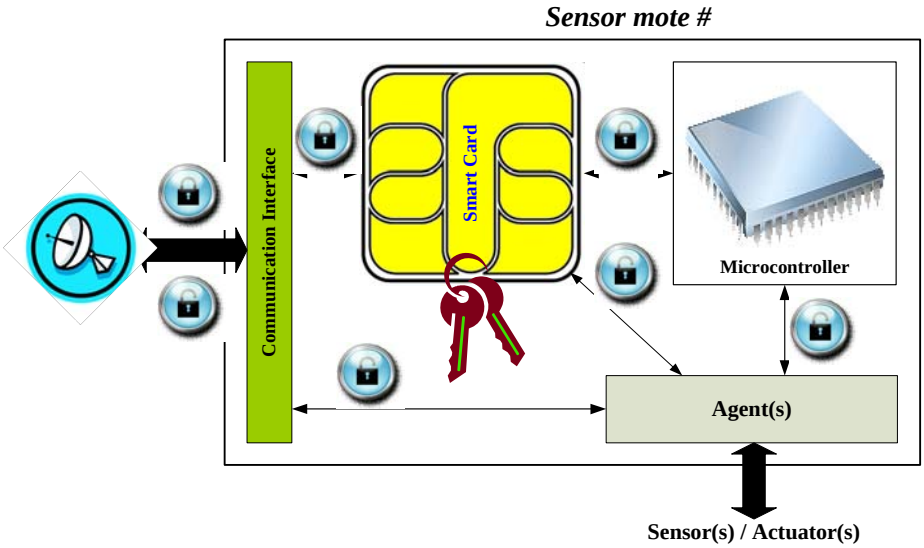

Figure 8: Architecture scheme.

As shown in Fig. 8, each sensor mote consists of two interfaces (communication with sensor(s) or actuator(s) and wireless communication with the outer world),

microcontroller, agent(s) and a smart card. Each sensor mote is a separate unit, which communicates with other motes in the wireless sensor network, and performs the programmed tasks, which are either pre-programmed or delivered via other agents. The microcontroller is responsible for enabling the measurement and agent process, and sends data or code to the smart card. The smart card in each note includes a secret key (either for symmetric or asymmetric cryptography) and performs the encryption and decryption of data or code (agents). The secret key is stored in secret memory and could not be copied out.

The agents have two main tasks – the first one is common for wireless sensor networks, i.e. to ensure measurement, data transfer and storage and are responsible for feedback to actuators. If one node is inaccessible, agents restore the previous functionality of the whole network (ad-hoc solution), whereas the missing node (mote) is compensated through other sensor motes (bridge). The second task of agents could be the security assurance. Each agent could be a heuristic toll, which tries to find attacks or intrusions tries and stops the "suspicious" mote or makes a bridge that the "suspicious" mote is jumped over. Such heuristics is joined with the possibility to reconstruct the wireless sensor network in ad-hoc principle.

The communication inside the mote is unsecured, i.e. not encrypted. The communication with the outer world is done either encrypted or as a plain text. Agents, which are used for heuristic analysis of the WSN security, travel in the network in opened form – they could not be enciphered. On the other hand, all data and commands are send in enciphered form, because the security of them could be ensured only if these data are resistant against attacks.

The significance is such solution is in contemporary wireless sensor networks that are used in security relevant regions – (petro/bio)chemical, medicine and military Industrial solutions. The usage of wireless data transfer in such branches is pushing ahead by big industrial companies. Such solutions often do not offer any securing of data transfer. These networks are able to make an ad-hoc reconfiguration, but are not able to detect intruders and attacks. In addition, applications with very high security requirements (nuclear industry, poisons etc.) do not allow to send information from separate motes in an open form, because only the information from these nodes could be used for extensive attack, based on social engineering.

5 Conclusion

This article outlines the possible usage of cryptographic protocols for securing of a wireless sensor network. Such cryptographic protocol could be combined with an agent, which is a part of a program code. The computational power for security tasks and for the storage of a cryptographic key could be ensured by using of a smart-card, which offer enough computational power and is prepared for secure storage of a cryptographic key.

Acknowledgement

This research has been done under the support of the grant "*Security-Oriented Research in Information Technology*", MSM0021630528 (CZ).

References

[Amin 2008] Amin, F., Jahangir, A. H., Rasifard, H.: *Analysis of Public-Key Cryptography for Wireless Sensor Networks Security,* Proceedings of world academy of science, engineering and technology, Volume 31, 2008.

[And 1972] Anderson, J.P. *Computer Security Technology Planning Study*, ESD-TR-73-51, vol. I, ESD/AFSC, Hanscom AFB, Bedford, Mass., October 1972 (NTIS AD-758 206).

[ATM] ATMEL: ATMega128 Datasheet: 8-bit AVR Microcontroller with 128K Bytes In-System Programmable Flash. http://www.atmel.com/dyn/resources/prod_documents/doc2467.pdf, 2008.

[Bra 1987] Bratman, M.E.: *Intention, Plans and Practical Reason*, Harvard University Press, Cambridge, MA, 1987.

[Che 2002] You-Chee, T., Sheng-Po, K., Hung-Wei, L., Chi-Fu, H.: *A Mobile-Agent Approach for Location Tracking in a Wireless Sensor Network,* International 1st Computer Symposium, 2002.

[EMV 2008] EMVCo: EMV Integrated Circuit Card Specifications for Payment Systéme, Book 1-4, 2008.

[FIP 2008] *Foudation for Intelligent Physical Agents*, http://www.fipa.org.

[FIPS 140-1] Security Requirements for Cryptographic Modules, FIPS PUB 140-1, Federal Information Processing Standards Publication, National Institute of Standards and Technology, U.S. Department of Commerce, January 11, 1994.

[FIPS 140-2] Security Requirement for Cryptographic Modules, FIPS PUB 140-2, Federal Information Processing Standards Publication, National Institute of Standards and Technology, U.S. Department of Commerce, May 25, 2001.

[GPK 2001] Gemplus: GPK Reference Manual, 2001.

[Han 2005] Hanáček, P.: *Problems of Security in Ad Hoc Sensor Network,* Modelling and Simulation of Systems, 2005.

[Lev 2006] Levis, P.: *TinyOS Programming*, 2006, http://csl.stanford.edu/~pal/pubs/tinyos-programming.pdf.

[Lia 2005] Chien-Liang, F., Gruia-Catalin, R., Chenyang, L.: *Rapid Development and Flexible Deployment of Adaptive Wireless Sensor Network Applications*, Proceedings of the 24th International Conference on Distributed Computing Systems, Columbus, 2005, pp. 653-662.

[Nas 2007] Nassu, B.T., Nanya, T., Duarte, E.P.: *Topology Discovery in Dynamic and Decentralized Networks with Mobile Agents and Swarm Intelligence*, Proceedings of 7th ISDA, IEEE Computer Society, Washington, 2007, pp. 685-690.

[Per 2004] Perring, A., Stankovic, J., Wagner, D.: *Security in Wireless Sensor Networks*, Communications of the ACM, Vol. 47, No. 6, 2044, p. 53, ISSN 0001-0782.

[Pur 2008] Purelink Technology: http://www.purelink.ca/.

[QiH 2001] Qi, H., Iyengar, S. S., Chakrabarty, K.: *Multi-resolution data integration using mobile agents in distributed sensor networks*, IEEE Transactions on Systems, Man, and Cybernetics Part C: Applications and Reviews, vol. 31, no. 3, 2001, pp. 383-391.

[Rankl 2003] Rankl, W., Effing, W.: Smart Card Handbook, Third Edition, John Wiley & Sons, Ltd, 2003.

[Rao 1995] Rao, A. S., Georgeff, M. P.: *BDI Agents: From Theory to Practice*, Proceedings of the 1st Conference of Multiagent Systems, 1995.

[Rao 1996] Rao, A.: *AgentSpeak(L): BDI Agents speak out in a logical computable language*, Agents Breaking Away, Lecture Notes in Artificial Intelligence, Vol. 1038, Springer-Verlag, Amsterdam, 1996.

[Sma 2008] SmartCardBasics: http://www.smartcardbasics.com/.

[Tir 2008] Tiresias: http://www.tiresias.org/.

[Tru 1985] Department of Defense: *Trusted Computer System Evaluation Criteria*, DoD 5200.28-STDm December 1985, US Department of Defense, December 26, 1985.

[Wal 2006] Walters, J.P., Liang, Z., Shi, W., Chaudhary, V.: *Wireless Sensor Network Security: A Survey*, In: Yang Xiao (Eds.): Security in Distributed, Grid and Pervasive Computing, Auerbach Publications, CRC Press, 2006, p. 50, ISBN 0849379210.

[ZigBee 2008] http://www.zigbee.org/en/index.asp, 2008.

[7816-1] International Standardization Organization (ISO).: Integrated circuit(s) cards with contacts - part 1: physical characteristics. ISO/IEC 7816-1, 1998.

[7816-3] International Standardization Organization (ISO).: Integrated circuit(s) cards with contacts - part 3: electronic signal and transmission protocols. ISO/IEC 7816-3, 1997.

[7816-4] International Standardization Organization (ISO).: Integrated circuit(s) cards with contacts - part 4: interindustry commands for interchange. ISO/IEC 7816-4, 1995.