Extending Extensible Authentication Protocol over IEEE 802.15.4 networks

Marcin Piotr Pawlowski*,†, Antonio J. Jara† and Maciej J. Ogorzalek*
*Faculty of Physics, Astronomy and Applied Computer Science
Jagiellonian University, Krakow (Poland)
†Institute of Business Information Systems
University of Applied Sciences Western Switzerland (HES-SO), Sierre (Switzerland)
Email: marcin.pawlowski@uj.edu.pl, jara@ieee.org, maciej.ogorzalek@uj.edu.pl

***Abstract*—Internet of Things is one of the major evolutions in the Internet, after the Web. Bringing the Internet into our physical world and making it present everywhere. This evolution is also raising challenges in issues such as privacy, and security. For that reason, this work is focused on the integration and lightweight adaptation of existing authentication protocols, which are able also to offer authorization and access control functionalities. In particular, this work is focused on the Extensible Authentication Protocol (EAP). EAP is widely used protocol for access control in local area networks such Wireless (802.11) and wired (802.3). This work presents an integration of the EAP frame into IEEE 802.15.4 frames, demonstrating that EAP protocol and some of its mechanisms are feasible to be applied in constrained devices, such as the devices that are populating the IoT networks.**

***Keywords*-Internet of Things; Authentication; Security; IEEE 802.15.4; EAP; 802.1X**

I. Introduction

Internet of Things (IoT) is one of the disrupting technologies that will change the future of our lives. The prognoses are that till the year 2020, billions of new devices will be connected and deployed around us [1]. Myriad of new devices will be responsible for our well being at home, work places, cars, and cities. For that reason, it is very important that technologies behind the Internet of Things should be reliable, easy-to-use, and safety.

Security in the IoT is one of the major pending challenges. Security is a strong challenge due to, on one hand, the constrained capabilities in terms of communications, memory, and computation, due to the needs of a low cost optimization in order to satisfy the requirements to make feasible a high volume production and mass deployment of IoT devices. On the other hand, most of the IoT devices will be battery operated through all their lifetime so energy efficiency requirements are very high. In addition to the constraints itself, IoT presents new challenges for the bootstrapping and commissioning of their functions and security credentials due to the raising high number of newly deployed devices, where theirs maintenance is infeasible to be performed by humans any more [2].

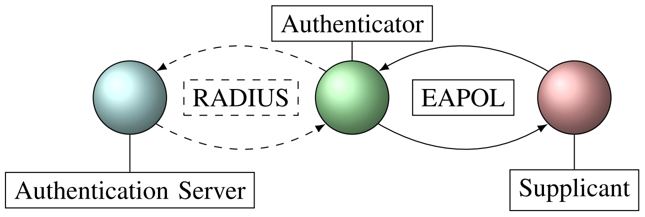

Figure 1. Schema of IEEE 802.1X secured networks.

For these reasons, this work is focused on analysing the feasibility of existing authentication, authorization and access control protocols for the emerging IoT networks. In particular, this work will be focused on 802.1X and associated technologies such as Extensible Authentication Protocol (EAP), Remote Authentication Dial In User Service (RADIUS), and EAP over Local Area Networks (EAPOL).

The rest of the paper is organized as follows. Section II consists of background knowledge of authentication protocols used in our research. Section III describes our testbed environment. In Section IV we present our approach to minimization of EAPOL overhead. We present tests and results of our apporach in Section V. Section VI addresses our next research steps and finally the paper is concluded in Section VII.

II. Internet Authentication Architecture

The most commonly used authentication protocol in Wireless Local Area Networks (WLANs) is EAP. EAP is only a part of the bigger infrastructure initially specified in IEEE 802.1X standard. Additional mechanisms needed for proper authentication service are shown on Figure 1. A comprehensive authentication mechanism consists of three services (*Authentication Server*, *Authenticator* and *Supplicant*), and two protocols that are transporting EAP frames (*RADIUS* and *EAPOL*). This section presents the basis about the mentioned services.

A. Authentication Server

Authentication Server is a service responsible for generating cryptographic challenges and calculating correctness of the cryptographic responses. In general it is standalone server located in secured part of the infrastructure but it also might be integrated with the Authenticator. A RADIUS server is a common example of the Authentication Server.

B. Authenticator

Authenticator is a service running on the device that is on the edge between secured and unsecured parts of the network. It is responsible for mediating between Supplicant and Authentication Server. In Wireless Local Area Networks the Authenticator is normally located on the Access Point.

C. Supplicant

Supplicant is a service located on the device that is trying to connect to secured part of the network. It is responsible for initiating the authentication procedure and responding to the request messages. In the IEEE 802.11 networks the Supplicant is running on the client device.

D. Extensible Authentication Protocol (EAP)

EAP is an authentication framework defined in RFC 3748 [6]. This standard does not define authentication mechanism but provides common functions for the authentication algorithms, which are called EAP-Methods.

Figure 2 presents the EAP packet datagram, which is composed of three mandatory fields (marked as green) and an optional data field. First octet, the *Code* indicates the type of the EAP packet. It can be set to *Request (1)*, *Response (2)* and *Success (3)* or *Failure (4)*. Next byte is *Identyficator* which is a counter that is incremented in every round of the communication. Last field is dedicated for the definition of the *Length* of the Payload. If *Code* is either *Request* or *Response* there is an additional mandatory field *Type* that defines the type of EAP authentication and dictates how the Payload should be processed. The *Type* field might be set to one of the following *Identity (1)*, *Notification (2)*, *NAK (3)*, *MD5-Challenge (4)*, *TLS (13)* or different value defined in additional specifications.

E. Extensible Authentication Protocol Over Local Area Networks (EAPOL)

EAPOL is a mechanism that encapsulates EAP messages and transfers them between Supplicant and the Authenticator [3] over the link layer. This feature is highly important, since EAPOL allows the transport and exchange of credentials, even when the access to the network at the network layer (IP layer) has not been yet granted.

Figure 2 presents IEEE 802.11 EAPOL frame with EAP Payload [4]. All mandatory fields consume only 4 bytes of the frame. First byte, defines the *Version* of the EAPOL frame, it can be set to 1 (which means that the frame is compliant with IEEE 802.1X-2001 revision) or it might be set to 2 (frame compliant with IEEE 802.1X-2004 revision). Second byte is responsible for the definition of *Packet Type*.

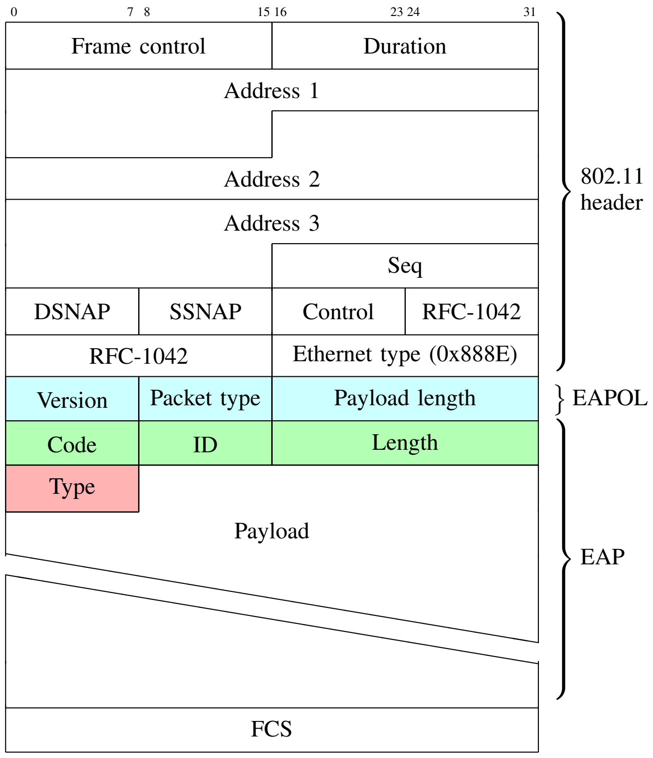

Figure 2. 802.11 EAPOL frame with EAP Payload.

There are only five different packet types, *EAP Packet (0)*, *EAPOL Start (1)*, *EAPOL Logoff (2)*, *EAPOL Key (3)* and *EAPOL Encapsulated Alert (4)*. Last two bytes are devoted for the definition of the *Length* of the EAP Payload.

F. Remote Authentication Dial In User Service (RADIUS)

RADIUS provides authentication, authorization and accounting (AAA) services [7]. In the IEEE 802.1X standard it is working as the Authentication Server and provides challenges and evaluation of the responses from the Supplicant. As shown on Figure 3 RADIUS packet consists of four mandatory fields, marked as green. First field, *Code* defines the type of the packet, it can be set to *Access-Request (1)*, *Access-Accept (2)*, *Access-Reject (3)* and *Access-Challenge (11)*. Next byte is *Identifier* responsible for matching requests with responses and detecting duplicates. Third field *Length* consist of two bytes and indicates the size of the packet. Last field is the *Authenticator*, 16 bytes of special MD5 checksum of the packet that verifies validity of the RADIUS requests and responses. There can be also optional *Attribute Value Pairs (AVP)* fields that are required if the *Code* is set as *Access-Request* or *Access-Challenge*. In the AVP fields encapsulation of the EAP packets is performed [8]. Structure of *Attribute Value Pairs* is simple and only consists of *Type*, *Length* and *Payload*. The AVP *Type* might be set to the *User-Name (1)* if the name of the user would be

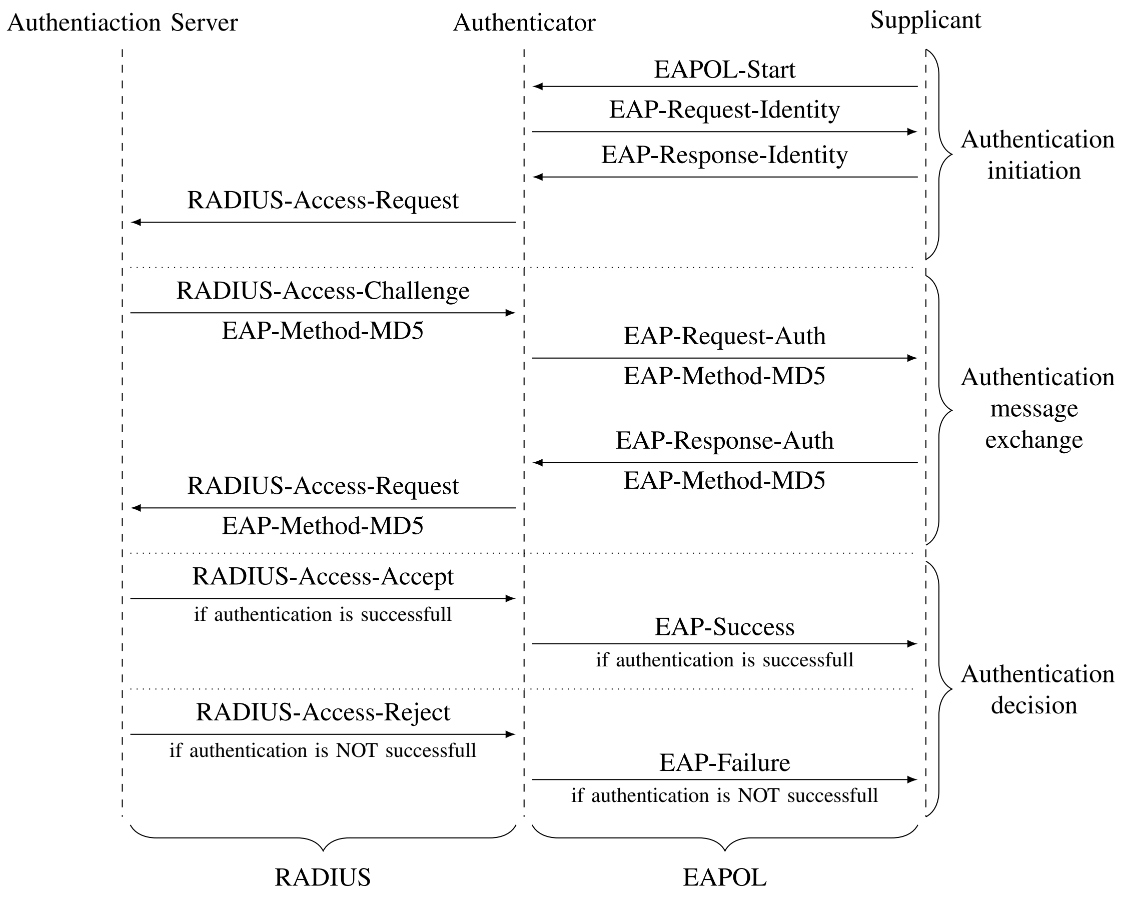

Figure 4. Example IEEE 802.1X message exchange with EAP-Method-MD5 as an authentication mechanism.

sent or set to *EAP-Message (79)* with accompanying mandatory *Message-Authenticator (80)* AVP if the EAP packet would be encapsulated. RADIUS packets are delivered by the UDP protocol to the RADIUS server port 1812 using general IP communication.

G. IEEE 802.1X

Figure 4 presents example of IEEE 802.1X message exchanged with MD5 authentication mechanism. In general, we can distinguish three different phases during IEEE 802.1X message exchange. [9]

First phase consists of just four communicates, which we have called *Authentication Initiation* phase. At the beginning Supplicant sends *EAPOL-Start* message to the Authenticator that informs Authenticator to start the EAP authentication procedure. After the reception the Authenticator by sending the *EAP-Request-Identity* asks Supplicant to identified itself. Supplicant responds with the *EAP-Response-Identity* packet with its identity string inside. Authenticator encapsulates the *EAP-Response-Identity* in *RADIUS-Access-Request* datagram and forwards it to the Authentication Server. Then RADIUS server checks the validity of identity of the Supplicant and if everything is correct then proceeds to the next phase, otherwise it responds with the *RADIUS-Access-Reject* message.

Second phase, *Authentication message exchange* in this example is only four communicates long, due to the fact that EAP-Method-MD5 was used as the authentication mechanism. With different EAP-Method the number of exchanged messages is bigger (in general). During this phase the negotiation and execution of authentication mechanism is performed. This starts the negotiation of the authentication mechanism, the *RADIUS-Access-Challenge* includes EAP-Method-MD5 challenge to which the Supplicant should respond. The Authenticator strips the RADIUS packet and forwards the challenge in the *EAP-Request-Auth* message to the Supplicant. If the EAP-Method would be not acceptable by the Supplicant the *EAP-Response-NAK* message should be sent, after that another EAP-Method will be selected by the Authentication Server. In this example Supplicant accepts the Challenge, prepares the EAP-Method-MD5 re-

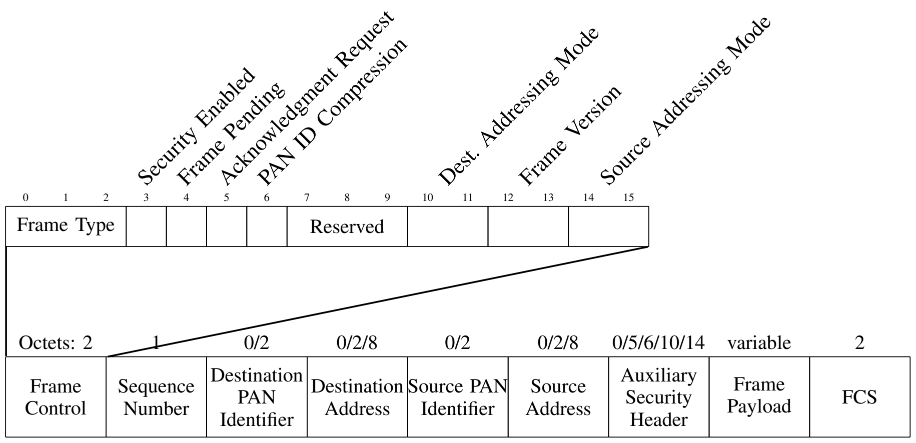

Figure 5. IEEE 802.15.4 MAC frame format.

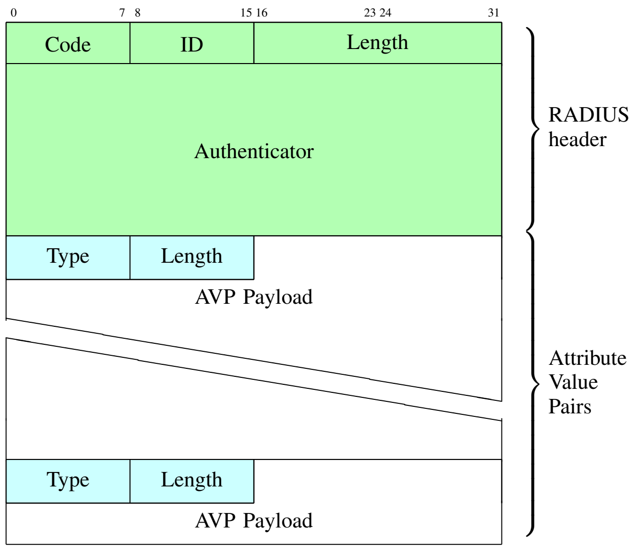

Figure 3. RADIUS packet datagram

sponse and sends the answer back to the Supplicant in the *EAP-Response-Auth* communicate. The Authenticator relays the response to the Authentication Server in the *RADIUS-Access-Request* packet. Which ends the authentication message exchange phase.

Last phase is *Authentication decision* during which the Authentication Server decides if the Supplicant should be allowed to have granted access to the protected resources. If the authentication was successful the Authentication Server sends *RADIUS-Access-Accept* message to the Authenticator. The Authenticator starts the access granting procedure and sends *EAP-Success* message to the Supplicant. If the authentication failed, analogical procedure is performed and the access is rejected, which is indicated by the *EAP-Failure* message sent to the Supplicant.

III. Testbed

A. Hardware

All of our tests were performed on the Tmote Sky boards with MSP430 MCU and IEEE 802.15.4 wireless links working in 2.4 GHz frequency. One Tmote was connected to regular PC through USB port and acted as an Access Point. Other Tmotes were acting as a Clients and were attempting to authenticate and to connect to the network.

B. Software

For the tests we have used three different Contiki OS applications.

1) native-border-router: is an application that works as a border router between IEEE 802.15.4 network and regular Internet. It connects itself through tun/tap interface with the Ethernet interface of the host machine and communicates through SLIP protocol with the IEEE 802.15.4 capable device connected with the host machine through the USB port. It relays packets between those networks. We have added new routines that allowed the transfer and encapsulation of the EAP packets to the RADIUS server protocol and the way back.

2) slip-radio: is an application that runs on the IEEE 802.15.4 capable device connected with the host machine through USB port. It communicates with the *native-border-router* through the SLIP protocol and relays IEEE 802.15.4 frames. We have changed its MAC driver to the EAP capable one and implemented counterpart functionalities that allowed the transmission of EAP packets to the native-border-router.

3) sky-websense: is an application that runs solely on the IEEE 802.15.4 capable device and it consist of small http server that is accessible through IPv6 protocol. In this case we have made minimal changes and only switched its MAC driver to EAP capable one.

	Phase 1	Phase 2	Phase 3
EAPOL	5	1.0724	0.5321
SEAPOL	5	1.0726	0.5223

Figure 7. Estimated delays of the authentication phases.

	Original	NullMAC	EAPOL	SEAPOL
slip-radio	33418	30618	33380	33110
sky-websense	48832	45390	49144	48808

Figure 8. Number of bytes of the applications in the Flash memory of the IoT device.

C. Implementation

We have successfully implemented our approach in the Contiki OS [1].

In order to optimize the memory footprint and considers a MAC layer without IEEE 802.15.4 design constrains, we have chosen minimalistic NullMAC driver and extended it to support two separate roles of the Supplicant and the Authenticator. Next we have added minimalistic EAP layer with only EAP-Method-MD5 authentication mechanism to verify the correctness of our design. At the end we have implemented minimal RADIUS communication layer to support the message exchange with the FreeRADIUS server through the IPv6 protocol.

During the tests we were able to successfully authenticate the Supplicant devices with regular FreeRADIUS server using EAP-Method-MD5 authentication mechanism. This shows that EAP protocol is capable to run in the highly constrained environments of the Internet of Things.

IV. Optimizing EAPOL protocol

Our main goal was to design mechanism to deliver EAP messages between IoT devices with as minimal footprint as possible. To accomplish that we have designed Slim Extended Authentication Protocol over Low-Rate Wireless Personal Area Networks (SEAPOL) that utilized just three bits of the reserved part of the Frame Control field of the IEEE 802.15.4 MAC frame header. In comparison to the regular EAPOL that is using 2 bytes of the *Ethernet Type* and additional 4 bytes for the EAPOL header, we have achieved significant savings. We have also used Data Frame Type and requesting acknowledgements for every transmission. Figure 6 presents our approach.

[1]Contiki OS by SICS: http://www.contiki-os.org/

	Original	NullMAC	EAPOL	SEAPOL
slip-radio	75770	71906	75552	75189
sky-websense	96789	91293	97896	96789
native-border-router	413177	413177	453157	453157

Figure 9. Code size of the applications binary files.

V. Tests and Results

First test we have performed was to measure the sizes of application that was modified during implementation and optimization phases. We have measured the binary file size and the allocation in the flash memory of the Tmote devices. Due to the fact that we have modified NullMAC and NullRDC drivers we will be comparing our results with this drivers instead of the original ones. In table 9 and table IV we have presented sizes of different versions of the implementations.

Our modifications to the *native-border-router* increased its size around 9.8% that is additional 39980 bytes. It was mostly due to the fact of implementation of RADIUS communication mechanisms. Size of *native-border-router* is less important because it usually runs on much more powerful devices that could handle a lot more.

Slip-radio modification added 2762 bytes to the 30618 bytes of flash memory of the NullMAC/RDC version which stands for 9% increase. Due to our effort of optimizing EAPOL protocol and implementation of SEAPOL we have successfully reduced the size of additional 270 bytes (9.8%) and decreasing the difference to the 8.1% of the size of the reference version.

Modifications of the *sky-websense* application introduced overhead of 3754 which is 12.2% more used flash memory. We ware able to decrease this size for 336 byte (8.9%) and achieve 11.1% increase of flash memory usage by switching standard EAPOL implementation to more optimized SEAPOL.

Last test was performed by measuring delays of the authentication phases. We have listened to the traffic on our RADIUS server and measured the timestamps of packet arriving and departing from our network. To measure the completion of Phase 3 (*authentication decision*) we have made small modification to the supplicant and forced resending last *Access-Request* packet. This modification allowed us to estimate the delays of all phases. From our measurements we were unable to determine the differences between EAPOL and SEAPOL implementations. Results presented on table 7 should not be considered as significant due to influence of various external factors on our measurements.

VI. Future work

Our effort to support the EAP protocol for the Internet of Things will be continued. For the future work we are planning to minimize the overhead of the EAP protocol by the header compression. Next we will focus on the optimizations of different EAP-Methods. Last part of the research will be devoted to finding cryptographic algorithms that are more optimized for the needs of the IoT devices and would serve best as the EAP-Methods.

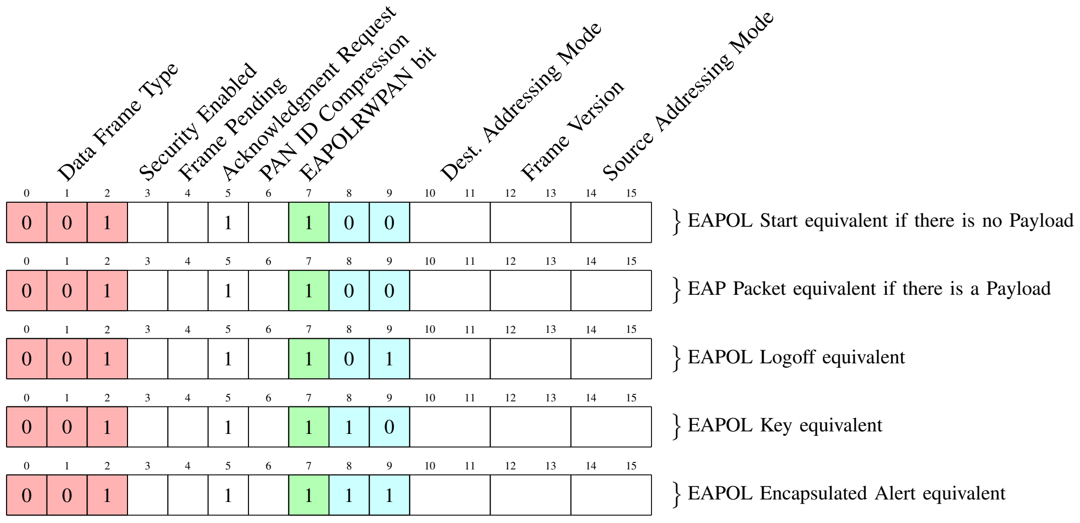

Figure 6. IEEE 802.15.4 Frame Control field modifications to support Slim Extensive Authentication Protocol over Low-Rate Wireless Personal Area Networks.

VII. Conclusion

This work has presented and compared two approaches to support EAP in the IoT networks. We have successfully implemented and tested EAPOL in the IEEE 802.15.4 network of TelosB devices running under control of the ContikiOS. Then we have introduced SEAPOL, an optimized version of EAPOL. We have shown that our approach gives around 9% of memory savings in comparison to the orginal EAPOL implementation. This work showed that it is possible to use EAP protocol as an authentication mechanism in very constrained environments of the Internet of Things.

Acknowledgement

The authors would like to thank to the HES-SO and the Institute of Information Systems funding and support. This project has been supported by the Swiss national government through the Sciex-NMSch (Scientific Exchange Programme between Switzerland and the New Member States of the EU) with the project code 13.121, named BASTION "Bootstrapping, Authentication, Security and Trust for the Internet of Things Networks". In addition, the authors would like thank to the European Project "Universal Integration of the Internet of Things through an IPv6-based Service Oriented Architecture enabling heterogeneous components interoperability (IoT6)" from the FP7 with the grant agreement no: 288445. Finally, this project has been also partially supported by the grant of the Polish Ministry of Science and Higher Education number 7150/E-338/M/2013.

References

[1] Pretz, Kathy. "The next evolution of the internet". IEEE Magazine The institute, 50(5), 2013.

[2] Jara, Antonio J., Ved P. Kafle, and Antonio F. Skarmeta. "Secure and scalable mobility management scheme for the Internet of Things integration in the future internet architecture." International Journal of Ad Hoc and Ubiquitous Computing (3)13, 228-242, 2013.

[3] IEEE LAN/MAN Standards Committee. "IEEE Std 802.1 X-2004 (Revision of IEEE Std 802-1x-2001)." IEEE Standard for Local and Metropolitan Area Networks, Port-Based Network Access Control,(Dec. 13, 2004, IEEE Computer Society, LAN/MAN Standards Committee).

[4] LAN/MAN Standards Committee. "Part 11: wireless LAN medium access control (MAC) and physical layer (PHY) specifications. IEEE Std 802.11 i-2004." IEEE Comput Soc (2004).

[5] LAN/MAN Standards Committee. "IEEE Standard for Local and metropolitan area networks – Part 15.4: Low-Rate Wireless Personal Area Networks (LR-WPANs)". IEEE Computer Society (2011).

[6] Aboba, Bernard, L. Blunk, J. Vollbrecht, James Carlson, and Henrik Levkowetz. "RFC 3748: Extensible authentication protocol (EAP)." Network Working Group (2004).

[7] Rigney, C., S. Willens, A. Rubens, and W. Simpson. "RFC 2865: Remote Authentication Dial in User Service (RADIUS)." Internet Society (Jun. 2000) (2000).

[8] Aboba, Bernard, and Pat R. Calhoun. "RFC 3579: RADIUS (Remote Authentication Dial In User Service) Support For Extensible Authentication Protocol (EAP)". Internet Society, September 2003.

[9] Chen, Jyh-Cheng, Ming-Chia Jiang, and Yi-wen Liu. "Wireless LAN security and IEEE 802.11 i." Wireless Communications, IEEE 12, no. 1 (2005): 27-36.