SEC-HOME: A SECURITY-ENHANCED FRAMEWORK FOR SMART HOME ENVIRONMENTS

Ralf Salomon, René Romann

University of Rostock, Institute of Applied Microelectronics and Computer Engineering, Germany

Abstract

Research on smart environments, such as smart homes and smart offices, have recently received increasing attention. In addition to the design and functionality of those devices, current research also focuses on usability and security (privacy). This paper describes a framework for smart homes, called Sec-Home, that supports flexibility, different communication media, as well as simple means to ensure enhanced privacy. The first prototypical implementations indicate that for the chosen application types, the usage of asymmetric cryptography, e.g., public/private key pairs and digital signatures, is very suitable, since their computational demands are negligible; this is a bit contradictory to common sense in that symmetric encryption should be used because of the huge differences in the required computational footprints.

Keywords

smart home, security, remote control, home automation

Introduction

The Term “smart appliance” refers to rather small but still intelligent devices that are equipped with at least some sensors, actuators, some processing capabilities, and a network interface of some sort. A collection of smart appliances constitute an “ensemble” if they cooperate together in order to help users to accomplish their goals [1].

In addition to the development of such devices, particular emphasis is devoted to the development of various self-organizing mechanisms such that the ensemble supports the users in an unobtrusive way that does not require any user-based configuration efforts [2] [3].

The concept of smart appliances ensembles provides a rather generic *idea* in which *smart homes* constitute a more specific instance for further research and development activities. Normally, a smart home utilizes several intelligent actuators, such as light sources, garage doors, heaters, and window blinds, as well as several, potentially mobile controllers, such as switches, regulators, and smart phones. All these devices employ some *wireless* communication capabilities, such as WiFi, Bluetooth, infrared, cameras, and ultrasonics, with which they communicate with each other as well as with their users. Furthermore, the ensemble's major intent is to provide a usability as flexible, unobtrusive, and reliable as possible.

On the one hand, using wireless control elements is very convenient, state-of-the-art, and well-accepted by many users. On the other hand, however internetworking communication infrastructures are often subject to threats, such as the unauthorized usage of any of the devices. It would be quite bothersome, for example, if an unauthorized person would be erroneously able to open the front door or the garage. Some of these potential security threats are discussed in Sec. II.

For security issues, the pertinent literature [4] [5] [6] [7] [8] provides a large variety of different approaches. The usage of the transport layer security (TLS), for example, is one of the many options, which is currently explored by others [7] [8]. That research prefers an initial out-of-band authentication procedure in which both, the actuator and the control elements are paired with each other.

The usage of TLS, however, requires the installation of a proper communication stack, which is very much geared towards WiFi and the Internet. Internet-based communication is rather resource demanding, and thus more or less excludes low-cost and primitive media, such as infrared, ultrasonics, and Bluetooth; it should be noted though that the literature discusses some approaches allowing the inclusion of those media [7].

Therefore, Sec. III proposes a certificate-based, security-enhanced framework, called Sec-Home, which is characterized by a very low footprint and is thus suited

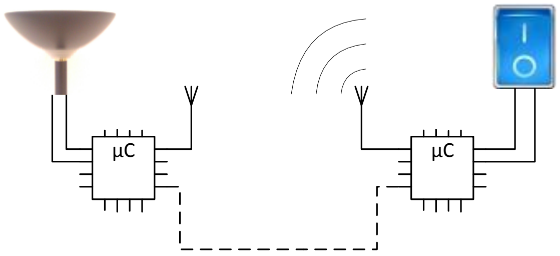

Fig. 1: Basic Setup, wireless and alternative wired connection (dashed line)

for smart homes and similar environments. Since the proposed Sec-Home framework utilizes some secrets, Sec. IV briefly describes the relevant administrative steps needed for system setup.

Finally, some of the major parts of the proposed Sec-Home framework have been realized in a laboratory at the University of Rostock. This setup is mainly for development and evaluation purposes. It consists of four lamps and a window blind and is described in Sec. V. The practical experiments reveal that on low-cost embedded systems, the execution of the RSA-algorithm takes about 10 milliseconds, which is *more* than acceptable for the targeted application. Section VI concludes this paper with a brief discussion.

Problem Description: Threats and Security

The General Setup

As an example, Fig. 1 shows two devices, a light source and a control button. Both devices employ some sort of processing unit as well as some communication interface by which they can communicate with each other. This communication can be established in a wireless or wire-based form. In both cases, it might well be that an additional server takes care of routing, accounting, and other administrative tasks.

For achieving the actual functionality, all the devices are equipped with an appropriate high-level communication protocol. The protocol might be application-specific, or, as will be probably done in the future, general purpose, such as device profiles for web services (DPWS) [5] [7]. By sending specific commands, the button might be able to switch on or switch off the associated light source.

The setup discussed above is for educational purposes only. A more general setup, might also contain window blinds, doors, and other general-purpose appliances.

Some very low-cost ("unintelligent") devices might not be technically or economically able to employ their own computational capabilities. These devices might be operated by the server, which runs the infrastructure, e.g., routing and accounting. For example, the server might implement an *n*-to-*m* relationship in which every of the *n* switches is linked or not linked to any of the *m* light sources. By means of this "configuration" server, a button might be able to switch on or switch off several light sources, since the server routes the incoming data packets to all the appliances that are indeed linked to that particular button. This server-based configuration approach is well known in the area of building automation [4] [9]. It should also be mentioned that this server might be working as a bridge that mediates between different network protocols, such as TCP/IP and Bluetooth or Infrared. An example of such network tool can be found in the literature [10].

For most readers, a generalization of this briefly discussed scenario is straight forward and only limited by the reader's imagination.

Security Issues

At first glance, even the very much simplified approach discussed above would be very convenient for most end users. Furthermore, the (smart) devices could be easily installed, since most of the required communication components already exist as part of the network infrastructure present in many households.

However, such an open environment would be a grateful target for severe security attacks of various sorts. An evil intruder might monitor, for example, the network traffic, and might thus obtain information about essential aspects, such as IP and MAC addresses, the structure of the data packets, the design of the employed communication protocol, and so forth. This is particularly true for communication media, such as Bluetooth, ZigBee, and the like, since they do not offer security mechanisms, such as firewalls and TLS connections. Once an intruder has obtained those details, he or she can operate all the devices, which potentially also may include the garage, heaters, the front and back doors, etc.

Related Research

Other research, especially [8], shows that there are multiple ways to secure the communication between two devices. This includes encryption at transport layer level, such as TLS or DTLS [11], but also encryption on message level, such as XML encryption.

Another approach uses a central authentication server together with asymmetric authentication mechanisms based on Kerberos and its extension SESAME [13]. It should be noted that the central authentication server in this approach needs to be online all the time whereas it does not need to be online within the proposed framework in Sec. III.

Enhanced Security: A First Proposal

The purpose of this section is to propose a *security-enhanced framework*, called Sec-Home, that is based on standard components and that is particularly suited for smart homes and similar environments. Two key issues of such environments are that the architecture is rather open and that most of their components require only very limited computational resources. The proposed framework can be described by the following rules:

1. A smart home employs at least one communication infrastructure, such as WiFi or a wired, IP-based network. In addition, other communication media, such as Bluetooth, ZigBee and Infrared might be employed.
2. All effectors, i.e., light sources, window blinds, and doors, as well as controls, i.e., buttons, dimmers, and smart phones, feature at least one suitable network interface. The capabilities are chosen such that the required devices can be communicating either directly or by means of an additional router, which might also be able to connect different communication media.
3. For every smart environment, the owner (or alternatively an administrator) selects a private-public key-pair that can be used in an asymmetric encryption scheme, such as RSA [14]. The private key is securely stored on a key server that should not go online for security reasons, whereas the public key is distributed among all devices that are part of the smart environment under consideration.
4. For every control, the key server selects a control-specific public/private key-pair that is being used with the very same asymmetric encryption scheme as mentioned in Rule 3.
5. For every control, the key server selects a device-specific certificate that includes at least the control's communication address (or name) and its public key, together with optional additional information.
6. The environment specifies a protocol with which the controls and actuators communicate with each other. Such a protocol contains at least the data that is to be transmitted to the actor. The protocol might furthermore specify some additional hand-shake messages.
7. Every control, called sender for simplicity, digitally signs every message by using its private key. It furthermore attaches its own certificate, which, among other things, contains its public key.
8. According to the previous rule, an actuator, called receiver for simplicity, receives messages that contain at least the actual data as well as the sender's certificate. Since the key server's public key is known to all participants, a receiver can authenticate the message's certificate and then in turn can authenticate the received data. If both steps are sound, the actuator might interpret the data and might be performing appropriate actions.

In addition to the rules presented above, the environment should first of all specify an algorithm that calculates the digital signature. For this purpose, the asymmetric encryption algorithm called RSA [14] together with the hashing algorithm SHA1 [15] are common choices. RSA requires two secrets, called public and private key. For the generation of the digital signature, the "signer" uses the private key. With the public key distributed to all participants, everyone can validate that the digital signature is indeed from the signer.

In the framework described above, the replay of previously recorded messages constitutes a serious security hole. This threat can be avoided by at least the following two options:

1. The communication partners maintain a sender-specific counter. Prior to sending a message, the sender (i.e., control) increments this value and includes the new value in every message. Then, the receiver (actuator) accepts only those messages that have a counter value larger than the previously stored one; replayed messages do not meet this condition, and can thus be easily detected.
2. The detection of replayed messages can also be done by the employment of a secure authentication protocol family known as challenge-response [16]. In short, this scheme basically works as follows: After the initiation of a control-to-actuator communication, the actuator responds with a challenge, usually a random number, which the control sends back to the actuator in a suitable form.

Administration: System Setup and Key Distribubtion

The framework as discussed above requires a few setup steps. It at least requires a "certificate server" for the generation of the systems private-public key pair as well as all the certificates. From a technical perspective, the generation of a key pair is quite easy and can thus be done by any PC. The important point, though, is that due to security reasons, it should not be online nor should it be remotely accessible. In addition, the framework requires the following installation steps:

1. Every new actuator has to be equipped with the public key of the certificate server (systems public key), in order to validate all of the relevant certificates.
2. Every control is shipped with an internal private-public key pair. The public key can be accessed and transferred to the certificate server. The private key is not externally accessible.
3. For every new control, the key server has to generate a certificate based on the public key of the control that signs at least the following data: control's public key, its id (e.g., a name tag), as well as application specific auxiliary data.

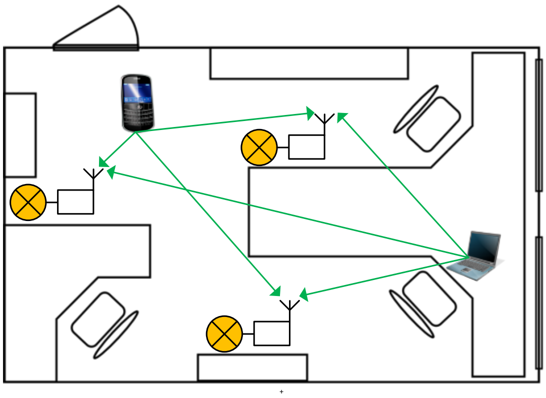

Fig. 2: Warnemünde SmartOffice

The installation procedure described above requires some additional (out-of-band) interfaces to transfer the public keys and certificates. Examples are an UART interface, a memory stick slot, a memory card slot, or the like.

Furthermore, a specific framework has to provide appropriate means for performing the required cryptographic algorithms. This can be realized, for example, by a processor and a suite of installed algorithms, such as OpenSSL which includes RSA and SHA-1 [17].

After installation of all needed certificates and distribution of the systems public key, the key server may be turned off and the systems private key should be stored in a safe place such as an USB stick.

Warnemünde's SmartOffice: A Concrete Case Study

Chosen scenario

For evaluation and development purposes, recent activities have developed a first test environment. This test environment has been integrated into a regular PhD students' office. Fig. 2 shows that it contains three light sources that are placed close to the desks. Each of the light sources employs a controller, a Raspberry Pi [18] microcontroller board, which features a WiFi communication interface. To overcome security risks due to message replay, the controller performs the response of the challenge-response procedure as well as the validation of the incoming messages.

Furthermore, the test environment employs a small number of laptops and Windows Phone based smart phones. All these devices run a graphical user interface with which the user can switch on, switch off and dim the light sources. Prior to transmitting the appropriate data packets, the graphical user interface initiates the challenge-response procedure and digitally signs all the data packets.

Performance Tests

For the first prototypical implementation, all devices have been equipped with a publically available software suite [17] that offers RSA and SHA1 with a key length of 4096 bits. These algorithms require about 5 megabytes of program memory, which is neglible compared to the memory size of 256 megabytes and more.

This first test uses a very simple text based protocol (e.g. "Lamp 1 on" for switching a lamp on). It should be noted that due to rule 6 from Sec. III the upper protocol is not part of the Sec-Home framework.

On the controller, the Raspberry Pi, the calculation of the SHA1 hash value and the decoding of the asymmetrically stored RSA signature data takes about 10 ms. Since every incoming message requires the validation of two signatures, the entire message validation consumes 20 ms.

The packet generation of the laptop and Windows Phone takes about 80 ms. The sum of the encryption and decryption times is less than 200 ms and has not been criticized by any of the test persons, since this delay was not noticed.

The practical tests have revealed one inconvenience that is caused by the operating system's shared library mechanism: the very first usage of the graphical user interface requires about 400 ms, since the operating system has to load the corresponding encryption library; after that first loading, the library resides in memory. In order to avoid this "problem," the user interface performs one pseudo operation of the SHA1 algorithm, in order to force the operating system to load the library.

Discussion

This paper has proposed a framework, called Sec-Home, which is intended to provide simple methods to enhance the security of a smart home. This framework is based on certificates and digital signatures. It merely requires the distribution of a global public key as well as a specific certificate on a per device basis. The chosen approach is not geared towards the usage of TCP/IP, and thus allows for an easy integration of other wireless communication media, such as Bluetooth, infrared, and ultrasonic.

A light-weight instance of the proposed framework has already been implemented and installed in a laboratory office. The first sample implementation consists of a Windows Phone, a PC, a few Raspberry Pi's, and a small number of light sources. The practical experiments have shown that both the generation and the validation of the signatures require less than 200 ms, which does not impose any inconveniences in this application type.

Among other things, future research will be dedicated to the integration of other types of light sources. This will be including high-voltage as well as dimmable torchieres. Furthermore, future research will be integrating Bluetooth devices as well as sensor nodes that are based on the Chipcon CC1010 chip. Also the integration of some window blinds will be completed in the near future.

Acknowledgement

Special thanks are due to the entire MuSAMA graduate school including all the PhD students as well as the supervisors for providing a very stimulating environment, which was essential for the development of the proposed framework. This work was supported in part by the DFG graduate school 1424.

References

[1] M. Weiser, "The computer for the 21st century," Scientific American, vol. 265, no. 3, pp. 94–104, Sep. 1991. [Online]. Available: http://www.nature.com/scientificamerican/journal/v265/n3/pdf/scientificamerican0991-94.pdf

[2] S. Poslad, Ubiquitous Computing: Smart Devices, Environments and Interactions. Queen Mary, University of London, UK: John Wiley & Sons Ltd., 2009.

[3] T. Kirste, Smart Environments. Berlin, Germany: Springer-Verlag, Aug. 2006, ch. 17, pp. 321–337.

[4] "ISO/IEC 14543: Information technology - Home Electronic System (HES) architecture."

[5] R. Chinnici, J.-J. Moreau, A. Ryman, and S. Weerawarana, "Web Services Description Language (WSDL) Version 2.0 Part 1," Jun. 2007. [Online]. Available: http://www.w3.org/TR/wsdl20

[6] (2014, Mar.) FHEM. [Online]. Available: http://fhem.de/fhem.html#Description

[7] C. Lerche, N. Laum, G. Moritz, E. Zeeb, F. Golatowski, and D. Timmermann, "Demo Abstract: uDPWS - The Devices Profile for Web Services for Resource-Constrained Devices," Feb. 2011.

[8] S. Unger, "How much security for switching a light bulb - The SOA way," 2012.

[9] "DIN EN 50090 VDE 0829: Elektrische Systemtechnik für Heim und Gebäude (ESHG)."

[10] (2014, Mar.) KNX Association - KNX IP. [Online]. Available: http://www.knx.org/de/knx-standard/ersten-schritte/

[11] E. Rescorla and N. Modadugu, "IETF RFC 6347: Datagram Transport Layer Security Version 1.2," Jan. 2012. [Online]. Available: http://tools.ietf.org/html/rfc6347

[12] V. Hernández, L. López, O. Prieto, J.-F. Martínez, A.-B. García, and A. Da Silva, "Security Framework for DPWS Compliant Devices," in Emerging Security Information, Systems and Technologies, 2009. SECURWARE '09. Third International Conference on, 2008, pp. 87– 92.

[13] J. Al-Muhtadi, M. Anand, M. D. Mickunas, and R. Campbell, "Secure smart homes using Jini and UIUC SESAME," in ACSAC '00. 16th Annual Conference, Dec. 2000, pp. 77–85.

[14] R. L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," Commun. ACM, vol. 21, no. 2, pp. 120–126, Feb. 1978. [Online]. Available: http://doi.acm.org/10.1145/359340.359342

[15] D. Eastake and P. Jones, "IETF RFC 3174: US Secure Hash Algorithm 1 (SHA1)," Sep. 2001. [Online]. Available: https://tools.ietf.org/html/rfc3174

[16] A. Beutelspacher, Kryptologie, 8th ed. Friedr. Vieweg & Sohn Verlag, Jul. 2007.

[17] (2014, Mar.) OpenSSL: About the OpenSSL Project. [Online]. Available: https://www.openssl.org/about/

[18] (2014, Mar.) Raspberry Pi. [Online]. Available: http://www.raspberrypi.org/

René Romann, M.Sc.
Institute of Applied Microelectronics and Computer Engineering
Faculty of Computer Science and Electrical Engineering
University of Rostock
18051 Rostock, Germany

E-mail: rene.romann@uni-rostock.de
Phone: +49 381 498-7255