Embedded Plug-In Devices to Secure Industrial Network Communications

Michael Heigl, Martin Schramm, Laurin Doerr, Andreas Grzemba
Deggendorf Institute of Technology
Deggendorf, Germany
{michael.heigl, martin.schramm, laurin.doerr, andreas.grzemba}@th-deg.de

***Abstract*—With Ethernet as an ubiquitous technology also emerging in industrial networks, security is becoming one of the most important aspects. Whereas IT-networks together with their security features in the office domain are subject of a dynamic change, networks in the industrial environment have a long operational time and therefore are more vulnerable to unauthorized access due to outdated hard- and software components. IT-threats such as Stuxnet are targeted to these areas and in the case of a failure of, e.g., a critical infrastructure this can directly harm the public security. Hence, in this paper a concept of an embedded plug-in device is proposed that secures the authentication and integrity of communication flows between two communicating network parties. A key focus of the development is the message authentication mechanism without neglecting the safety aspect imperative for industrial networks.**

I. Introduction

The rise of data communication and the fact of increasing networking in all aspects of life require more sophisticated security considerations for future electronic systems. Successful attacks are becoming more frequent due to a growing spectrum of attack vectors. Because of the omnipresent usage of automation in critical infrastructure it is a matter of fact that attacks directed to those systems can cause severe damage to public and economic [1].

The wish for continuous networking including connectivity to the internet required new ways of thinking. A possible approach is the combination of (formerly) proprietary systems with standard information technology such as the internet protocol family. This change to open technologies, for instance Ethernet or TCP/IP, enabled high transmission capacity and a device connectivity from field- to management-level [2]. The term Industrie 4.0 describes not only the combination of production processes with information technology but also the internetworking with classical IT office networks. Industry, energy, transport, private environment: the infrastructures are growing together [3]. The increasing degree of connection associated with a rise of cyber attacks creates a demand for sophisticated security solutions [4]. Security by design is not always possible. Reasons for this are the long operational time of plants and a plurality of interlocking legacy-infrastructures. Further IT-systems are often evaluated after the implementation and are supplemented with security solutions afterwards or replaced by new secure technologies, e.g., the Siemens Industrial Security concept [5]. Such solutions are accompanied with a high financial effort for operating companies and are therefore often not desirable.

Obviously, e.g., for industrial legacy-systems a different approach and methodology to secure communication between network parties must be elaborated. Thus, an embedded plug-in device will be described in this work that satisfies this requirement.

The remainder of the paper is organized as follows: Section II gives an overview of related concepts to this work. Section III briefly describes the requirements for secure communication. Section IV provides the proposed security concept including the algorithms and modules necessary for securing communication flows. Section V presents a proof of concept with a related evaluation and section VI finalizes the paper with a short conclusion and a glance at future work of the ongoing research work.

II. Related Research Work

A concept of a software independent solution for Ethernet LAN security is presented in [6] by introducing a hardware device utilized between a protected host and the network. Data through the network is transmitted under encryption and can only be decrypted by the intended recipients hardware device. An appropriate key management system is described in order to use a one-time-pad encryption based on DES. The concept of such a *SECURCOM* device has further been implemented and is restricted to be used within one security domain meaning the transmission of packets to different subnets is not possible.

The Open Source CRYpto-Bridge (OSCRYB) for secure Ethernet point-to-point industrial communications was presented in [7]. Two secure Ethernet networks each containing an Ethernet device can be connected through two OSCRYB devices. If Ethernet frames match predefined MAC addresses, they are processed including an on-the-fly encryption using AES performed through a SEC core. Otherwise the frames will be forwarded transparently. Further implementation of the SEC core to a scalable 128-bit AES-CM crypto-core is presented in [8].

A secure communication protocol implemented on ZYBO Zynq-7000 development boards has been presented in [9]. Input data from users are passed to a FPGA and then sent over the internet to the corresponding communication partner. The proposed technology separates the communication area (inside the ARM) from the cryptographic one (inside the FPGA)

ISBN 978-80-261-0602-9,

in which an AES algorithm along with a cryptograhpic co-processor for data integrity algorithms is applied.

The mentioned approaches encrypt the data on-the-fly which causes a considerable delay to the network traffic. Therefore the transmission of time critical data especially needed in industrial control applications is hardly feasible. Modifying packets through encryption and decryption could also be a safety related problem. If the processes fail, packets are not guaranteed to arrive deterministically at their predefined destination.

III. Security Concerns of Automation Communication

In this context two network participants communicate secure when the protection goals of exchanged information *Authentication*, *Confidentiality* and *Integrity* are complied. Within industrial automation the protection goals are almost identical but in contrast to conventional IT systems the priority is given to *Availability* and *Integrity*. IT security concepts, e.g., IPsec, SSL/TLS and VPN are mainly directed to the IT domain and cannot be trivially mapped to the industrial automation [10]. For instance, this is because some protocols require to meet timing constraints. In addition many embedded devices are applied only providing limited system resources in terms of memory and computational power.

Network participants intending to communicate securely must prove their identity which could be achieved by utilizing a public key infrastructure. Confidentiality could be provided for instance through encryption mechanisms and digital signature schemes or applied hash functions could preserve the integrity of information. However, deployed cryptographic techniques are computationally intensive and must not exceed the available device resources [10].

The proposed embedded plug-in device of this work provides integrity through HMACs for messages which are flowing transparently through the device. Identification is given through a combination of the Diffie-Hellman key exchange based on certificates with a platform integrity verification in conjunction with a direct attestation step. In order to not forfeit the availability criterion the embedded plug-in device is not modifying the original network messages. The HMAC algorithm is applied on-the-fly without interfering the actual information. Confidentiality, e.g., through encryption is not required for many protocols since it only delays the network messages which results in a considerable computation time which is not intended within the protocol standards.

IV. Proposed Security Concept

Ensuring authentic network communication is obtained through embedded plug-in devices. A simplified network communication contains two end devices (A and B) which are connected to a network. All devices are located in the same subnet. End device A and B having a communication relationship, meaning that A sends messages to B ($m_{\mathrm{A}\rightarrow\mathrm{B}}$) and B is sending messages to A ($m_{B\,\rightarrow\,A}$). The simple network relation is extended by two embedded plug-in devices C and D in a way that the messages between A and B can be authenticated. Figure 1 shows a basic scenario. The normal network communication stays untouched, meaning that A sends messages to B ($m_{A\,\rightarrow\,B}$) and B is sending messages to A ($m_{B\,\rightarrow\,A}$) without C and D interfering. On C and D an Ethernet bridge is implemented combining the two Ethernet interfaces to transparently forward network traffic. The main task of the embedded plug-in devices is to generate and verify HMACs for the messages of the end devices passing through the bridge interface, depending on defined policies.

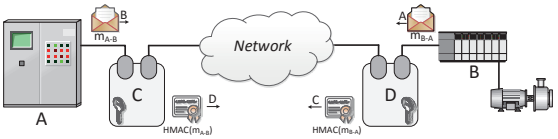

Fig. 1. Basic Scenario of the Conception

In this simple scenario C listens to the Ethernet interface to which A is connected. Depending on the rules defined in C Message Authentication Codes ($HMAC(m_{A\,\rightarrow\,B})$) will be generated for messages which will be sent from end device A to end device B ($m_{\mathrm{A}\rightarrow\mathrm{B}}$). The $HMAC(m_{A\,\rightarrow\,B})$ will be sent to and verified by D subsequently over the Ethernet interface connected to the network. For the creation of these authentication values both embedded plug-in devices C and D must first agree on a common secret. In the opposite communication direction D listens to the Ethernet interface to which B is connected and generates $HMAC(m_{B\,\rightarrow\,A})$ which will also be sent to and verified by the embedded plug-in device C.

A. Authenticated Boot

In order to ensure the integrity of hard- and software of the embedded plug-in device, an additional hardware-based security architecture as presented in [11] is applied as an integral part of the platform. Thus, a hardware chip, the Trusted Platform Module (TPM) will be used to secure the boot stages in which the process contains alternating measurement and execution flows on every consecutive boot step. Hence, a chain of trust can be established ranging from the first bootloader up to the application code.

B. Direct Attestation

Apart from verifying the authenticity of the communicating embedded plug-in devices, it is of interest if the communicating party is trustworthy regarding its hard- and software integrity. Since the mentioned authenticated boot process is used for verification of the integrity of hard- and software of an embedded plug-in device, this information can be used to authenticate its integrity to the communicating plug-in device. For this a special TPM key type is employed: The Attestation Identity Key (AIK). Apart from the unique Endorsement Key (EK) representing the identity of the platform the AIKs are created by the TPM and linked to the platform using certificates from the EK. The AIK can be used for signing purposes of the current

configuration (soft- and hardware) of the platform which is stored in terms of integrity fingerprints in the Platform Configuration Register (PCR) of a TPM.

With this concept two communicating embedded plug-in devices are able to verify its opposites hard- and software integrity. Based on this trust, a key agreement protocol can be performed as to obtain a common secret between two embedded plug-in devices in order to compute the Message Authentication Codes.

C. Key Establishment

The key establishment between two communicating embedded plug-in devices using an unsecured channel is performed by applying the well known Diffie-Hellman (DH) key exchange protocol as a basis. However, this scheme does not authenticate the entities involved and thus suffers from man-in-the-middle attacks [12]. Therefore, a key agreement scheme as shown in figure 2 has been implemented such that the authentication is anchored on certificates. The CA hierarchy in this example contains the certificates for two embedded plug-in devices $UserA$ / $UserB$ labeled as $cert_A$ / $cert_B$ which were certified by the root CA ($cert_{CA}$).

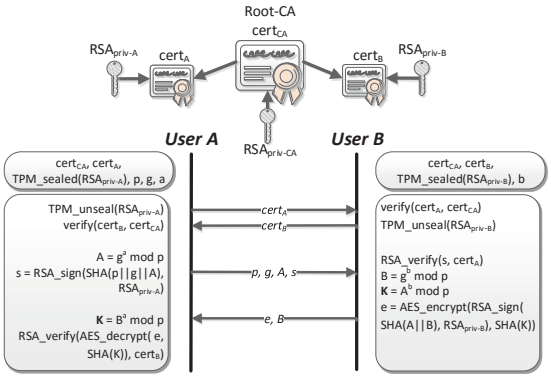

Fig. 2. Modified Diffie-Hellman Key Agreement

Before the actual key exchange starts, the two communicating parties verify each others certificates. For security reasons the private RSA signing keys are not desired to be stored on the embedded plug-in device's disk, since an attacker might compromise the platform and read out the private key. Thus, the TPM sealing functionality has been utilized in which the private key has been sealed previously in a trustworthy environment based on its hard- and software condition. The actual DH key exchange starts by $UserA$ computing the public DH key A using a large prime p, a generator g and a chosen private DH key a. Finding a large prime p and check whether g is a generator takes considerable time. Thus, a list of pre-calculated sets of p and g pairs are stored on $UserA's$ platform. The list gets updated every time a pair has been taken from the list. A concatenation of p, g and A will be hashed, e.g., using the SHA-256 algorithm. The result will be signed by $RSA_{priv\text{-}A}$, yielding the signature s. The set of parameters p, g, A and s is then sent to $UserB$, which will verify the signature under usage of $UserA's$ certificate $cert_A$. $UserB$ is able to compute his public DH key B and the common secret K by the received parameters A, p and his chosen private DH key b. In order to ensure that both communicating parties compute the same common secret, $UserB$ is encrypting a signature containing the concatenation of the public DH keys A and B with the common secret K yielding e. The cipher e and B will be transmitted to $UserA$ which is able to compute the common secret K. Whether the computation has been correct can be checked by first decrypting the cipher e using K and then verifying the result with $cert_B$. Since every time two embedded plug-in devices negotiate a new common secret K, Forward Secrecy (FS) is given.

D. Architecture

In the concept each embedded plug-in device is based on an architecture as shown in figure 3. Two Ethernet interfaces are bridged together (*br0*) in order to transparently forward network traffic from an end device (attached to *eth0*) to the network (attached to *eth1*) and vice versa. While packets are forwarded over the bridge, the *Capturing Module* is responsible to filter and process these packets according to predefined rules as well as forwarding them to the *HMAC Generation* module.

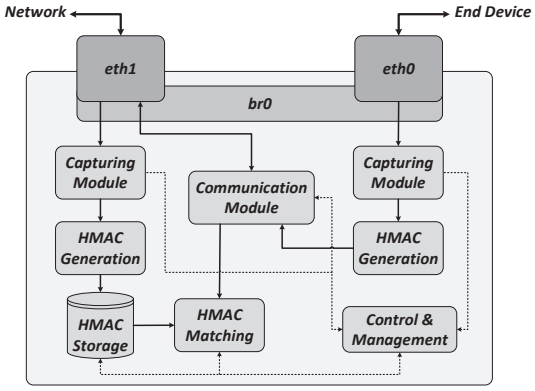

Fig. 3. Architecture of an Embedded Plug-In Device

HMACs will be computed by the *HMAC Generation* module using a previously negotiated common secret through the *Communication Module*. These HMACs are afterwards sent to the communicating embedded plug-in device within the network. Incoming packets from the network will be forwarded by the bridge *br0* but will also be subject to the *Capturing Module*. Then, the generated HMACs are stored in the *HMAC Storage*. If the hashes are received by the *Communication Module* from the communicating embedded plug-in devices within the network, they are supplied to the *HMAC Matching* module. It checks whether the computed HMAC and the received one are equal or not.

The *Control & Management* module is responsible for negotiating with communicating embedded plug-in devices how many packets and the point in time when packets shall be subjected to the HMAC generation. This reduces the network traffic and based on the frequency as well as number of computations the level of security can be adjusted. By utilizing for instance a time controlled counter approach, the module ensures that packets have not been dropped or replayed. Further

the module can be used to send information about packets to an Intrusion Detection System (IDS) or to inform an administrator about mismatching HMACs.

V. Proof of Concept

The embedded plug-in devices have been implemented using Terasic DE1-SoC Boards. An Ethernet extension card is used to connect two additional Ethernet PHYs through FPGA logic to the Hard Processor System (HPS) running an embedded Linux.

The bridging mechanism has been implemented by applying the bridging utility provided by the open source virtual switch *Open vSwitch*. The software supports forwarding of Bridge Protocol Data Units (BPDUs) frames. Thus, it could also be used together with layer-2 protocols, for instance PROFINET IO. *PF_RING* is a high speed packet capture library suitable for both packet and active traffic analysis and manipulation and provides a new socket type that can directly call a ring to read and write data [13]. Two of such rings are used to independently capture packets from the interfaces eth0 as well as eth1 and pass them to the Linux user space for HMAC generation/verification. The key establishment as well as the HMAC algorithm schemes have been implemented using functions from the open source library *OpenSSL*.

A simple scenario as shown in figure 1 has been set up. The ICMP and BACnet protocol in which end device A sends, e.g., simple BACnet packets to B has been used for proving the concept. A simple network router acts as a network infrastructure.

The forwarding latency of a packet can be determined by performing ping-commands between two devices using the ICMP protocol. In a first attempt a direct connection between the devices is applied and in a second an embedded plug-in device is interconnected. The difference in time for the ICMP packets to be sent out and received will be determined and its mean value of 10000 measurements yields the forwarding latency of 38.10 µs for a packet size of 98 bytes. A second approach utilizes the tool *SignalTap II Logic Analyzer*, in which the latency could be determined to 38.57 µs for the same packet size.

VI. Conclusion and Future Work

With upcoming Industrie 4.0 compliances, it is mandatory to develop security mechanisms for protecting industrial network infrastructures prone to security attacks. An approach to verify message integrity from authenticated network parties utilizing embedded plug-in devices has been delineated in this work in order to secure network connections.

Compared to similar approaches the presented concept allows security without impacting the actual traffic flow by decapsulating HMACs from their original packets. On-the-fly modification of network traffic, as is the case for encryption schemes, has a considerable delay impact on the transferred data. Encrypting 64 byte with the solution presented in [9] will take approximately 800 ns. Transmitting time and safety critical data could lead to safety issues if encryption or decryption fails. Thus, it is inevitable for industrial networks field level protocols, especially afflicted to meet timing requirements, e.g., PROFINET IRT, that the embedded plug-in devices are transparent in terms of not delaying packets. For a further work it is desirable to reduce the measured forwarding delay by implementing the bridging and hashing mechanism in hardware with the help of FPGAs since over 95 % of the delay is attributable to the applied embedded Linux.

The ongoing work will focus on extending the communication infrastructure to multiple embedded plug-in devices. Thus, automatically inserting and removing embedded plug-in devices in an existing infrastructure with regard to handling failures needs to be investigated. In the implementation of the proof of concept the *Control & Management* module has been neglected. The implementation of this will be content of future investigation. A feature for instance utilizing timestamps or a counter based approach is crucial in order to counteract, e.g., replay attacks in industrial networks.

Acknowledgment

This research work has been supported by the research project no. 16KIS0142 of the German Federal Ministry of Education and Research.

References

[1] Miyachi, T., Yamada, T., *Current issues and challenges on cyber security for industrial automation and control systems*, SICE Annual Conference (SICE), 2014

[2] Metter, M., Bucher, R., *Industrial Ethernet in der Automatisierungstechnik: Planung und Einsatz von Ethernet-LAN-Techniken im Umfeld von SIMATIC-Produkten*, Wiley, 2012

[3] DIN/DKE, *Deutsche Normungs-Roadmap IT-Sicherheit*, [Online], Available: https://www.dke.de/de/std/seiten/ normungsroadmaps.aspx, 2014

[4] Schramm, M., Leidl, K., Grzemba, A., *The Establishment of High Degrees of Trust in a Linux Environment*, Embedded World, 2012

[5] Siemens, *Totally Integrated Automation - Industrial Security*, [Online], Available: http://www.industry.siemens.com/topics/global/de/tia/industrial-security/seiten/default.aspx, 2015

[6] Hadjina, N., *SECURCOM - the security solution for Ethernet LANs*, Electrotechnical Conference, 2002

[7] Astarloa, A., Bidarte, U., Lazaro, J., Arias, J., Olaguenaga, E., *OSCRYB: Open Source CRYpto-Bridge for Secure Ethernet point-to-point Industrial Communications*, Industrial Electronics Society, 2007

[8] Astarloa, A., Lazaro, J., Jimenez, J., Cuadrado, C., *Scalable 128-bit AES-CM Crypto-Core Reconfigurable Implementation for Secure Communications*, Applied Electronics, 2009

[9] Marghescu, A., Svasta, P., *Secure Communication Protocol using Embedded Devices based on FPGA*, Electronics System-Integration Technology Conference (ESTC), 2014

[10] Granzer, W., Praus, F., Kastner, W., *Security in Building Automation Systems*, IEEE Transactions on Industrial Electronics, 2010

[11] Schramm, M., Grzemba, A., *Trustworthy Building Blocks for a More Secure Embedded Computing Environment*, Applied Electronics International Conference, 2011

[12] Nabil, M., Abouelseoud, Y., Elkobrosy, G., Abdelrazek, A., *Certificate-Based Authenticated Key Agreement Protocols*, Computer Applications Technology (ICCAT) International Conference, 2013

[13] Du, J., Liu, P., Elkobrosy, G., Abdelrazek, A., *Design and Implementation of Efficient One-Way Isolation System Based on PF_RING*, Multimedia Information Networking and Security (MINES) Fourth International Conference, 2012