Secure Vertical Integration for Building Automation Networks

Christian Reinisch, Wolfgang Granzer, Wolfgang Kastner *
Vienna University of Technology, Automation Systems Group
{cr,w,k}@auto.tuwien.ac.at

Abstract

In recent years, building automation systems have become a widely accepted technology with dedicated but stand-alone solutions existing for a variety of application domains. Data exchange across domain borders is – if at all – realized by interconnecting these networks at the management level. Thus, the next step is to realize end-to-end data exchange already at the device level. The resulting vertically integrated systems can host domain-spanning applications and make way for secured end-to-end communication among any group of devices. This is considered especially important if security-relevant information shall be exchanged.

This paper presents an adaptive security layer protocol architecture that is capable of operating on heterogenous networks. The modular framework is designed to support virtually any combination of network protocols and applications that meet the requirements best. Through a plugin-based approach, easy extension and reuse of existing protocol mechanisms is achieved.

1. Introduction

Building automation systems (BAS) allow automatic control of the environment, especially in the core domains heating, ventilation and air conditioning (HVAC), and lighting/shading. Today, also other systems such as access control and surveillance are considered for integration which widens the range for applications considerably. Sophisticated control tasks can be put into practice once information is shared across the different domains. For example, a window contact sensor (originally used in the security system) can also be used to (de-) activate the heating/cooling system.

However, it is characteristic of BAS that they are mostly realized using application-dependent subsystems, initially not designed for mutual information exchange. To provide the necessary means for information exchange, it has become common to interconnect the network segments by a common backbone. This integration approach is also referred to as *horizontal integration* [1].

*This work was funded by FWF (Österreichischer Fonds zur Förderung der Wissenschaftlichen Forschung; Austrian Science Foundation) under the project P19673.

Figure 1 shows a horizontally integrated network in which the subnetworks are realized using different network protocols. This underlying *heterogeneous network architecture* demands special attention once integration is planned. Therefore, gateway devices which are capable of assembling valid messages of both adjacent network protocols and also cater for a mapping of both protocols' semantics are used to establish an interconnection.

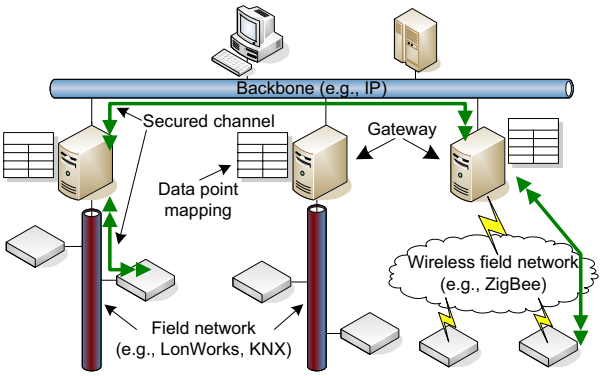

Figure 1. Gateway solution

However, a horizontal integration approach comes along with two main drawbacks. First, the gateways have to be capable of mapping all data points found in both attached networks that are of interest for the integrated applications. This requires large mapping tables to be stored and maintained. Additionally, a gateway is a critical component regarding internetwork communication reliability. In case of a gateway failure or mere misconfiguration, distributed applications can no longer communicate. Furthermore, gateways are prone to be targets of security attacks because the mapping tables concentrate the process data of the connected networks. Thus, gaining access to these tables provides a simple opportunity to manipulate the stored data points.

Second, comprehensive security (i.e., a secured channel guaranteeing data integrity, confidentiality, freshness and authentication) among two or more devices located in different network segments is hard to guarantee. The strict security requirements modern BAS are faced with stem from the fact that sophisticated applications also require security-relevant information to be shared. Devices with typically less strict or even no requirements regarding security (e.g., found in the HVAC domain) will ex-

978-1-4244-2350-7/08/$25.00 ©2008 IEEE.

change information with devices where security is probably achieved using proprietary mechanisms (e.g., a proprietary surveillance network). For example, a CCTV cam monitoring the entrance to a meeting room can be an input for the HVAC controller. The controller can then adapt its control strategy based on the number of people in the room.

As indicated in Figure 1, the only way to achieve comprehensive security in a horizontally integrated network is to secure each network segment separately. Secured channels are only established between a device and the corresponding gateway. At each gateway, the message has to be extracted and subsequently secured using the new protocol's mechanisms. The situation is further aggravated since not all protocols provide the same or even any security mechanisms. For example, both ZigBee 2006 [2] and BACnet Addendum g [3] provide mechanisms to ensure data integrity. However, they are based on different algorithms which renders them incompatible.

It is evident that these shortcomings have to be eliminated before a tighter integration can take place. A crucial factor for integration will be *end-to-end security* among any group of devices. This demands a unified and comprehensive security model that is applicable to any underlying building automation network.

This obvious need is the starting point for our work. In the next section, we will explore a solution that alleviates many drawbacks of horizontal integration. This feasible integration approach can adapt to the underlying heterogeneous network and enables comprehensive security. A stack featuring this as well as the reuse of adequate protocol mechanisms is presented in Section 3.

2. Vertical integration

As mentioned before an interconnection of heterogeneous networks using gateways has two main disadvantages that stem from the used gateways. Therefore, an apparent approach is to substitute the gateways by breaking-down the required functionality (including the necessary data point mapping) and distributing it directly to the attached devices. This leads to *vertical integration*, where the interconnection of network segments can now be accomplished using more simple routers instead of dedicated gateways (cf. Figure 2).

This router-based approach offers two main advantages. First, end-to-end security becomes possible. Using common security mechanisms, a router can simply forward the messages to the desired destination network[1] without facing the overhead of translation and reassembly of the secured network message. Therefore, devices located in different network segments are able to establish a secured channel directly between them. Second, the data point mapping between two network segments that had previously been implemented in the gateway is now

[1] It is assumed that the address information is secured in a way so that it can be processed by the routers (e.g., digitally signed but unencrypted).

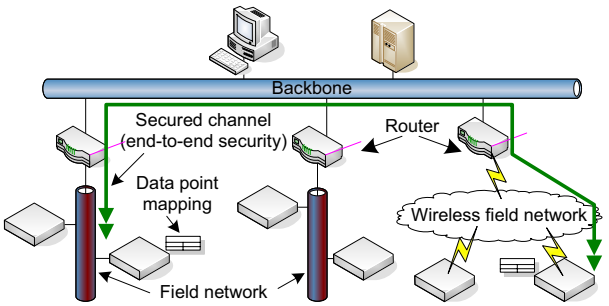

Figure 2. Router solution

distributed to the device. This significantly reduces the configuration effort since a device only has to bear the mapping of data points that are of particular interest for it. Therefore, generation and management of the mapping table does not imply much overhead. It is planned to further facilitate and (partly) automate this process with the help of management/engineering tools.

Another feature is that, as long as communication among one protocol is targeted, standard devices can be employed. The network is simply shared with the integrated devices. Likewise, only those devices that are required to communicate across heterogeneous network segments have to feature multiple protocols. This flexibility and the reduced (and more straightforward) mapping render vertical integration affordable.

3. Multi-protocol communication stack

Our approach towards vertical integration builds on a modular, plugin-based, multi-protocol communication stack. Its main features are support for a flexible combination of various network media, application layer protocols and user application types. Furthermore, mechanisms that guarantee comprehensive end-to-end security are included in the stack. The realization of our solution requires this advanced communication stack to be included in all devices that shall be able to communicate across protocol borders. The heterogenous protocol architecture is faced and used as input for the modular, plugin-based approach. As shown in Figure 3, the proposed stack is based on existing building automation protocols which are consequently extended with new services.

The stack is partitioned into three layers: the *Network-specific Layer (NSL)*, the *Security Abstraction Layer (SAL)* and the *Application-specific Layer (ASL)*. The NSL provides low level communication services used to transmit messages over the used network medium. The ASL implements the functionality of the application layer of the used building automation solution(s). The SAL abstracts the communication services of the NSL and offers generic secure communication services to the ASL. As all layers operate on plugins, easy extension is made possible.

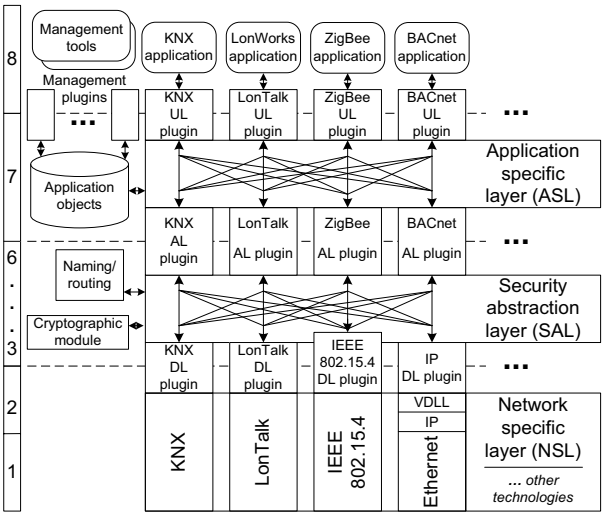

Figure 3. Multi-protocol stack

3.1. NSL

The NSL corresponds to layers 1 and 2 of the OSI reference model and provides access to the underlying network medium. Today, many different building automation standards exist which in turn support various network media. Each of them offers significant advantages regarding the physical characteristics, e.g., the high bandwidth of Ethernet or the free topology of KNX TP [4] and LonWorks TP [5]. Also various wireless technologies, with all their advantages and challenges, have emerged and thus merit attention. But protocols also differ at the data link layer. Only some offer native support for multicast (e.g., KNX, LonTalk) and even less provide security already at the data link level (e.g., IEEE 802.15.4 [6]).

To be able to satisfy the needs of BAS of all sizes and types, the proposed secure stack is not bound to any network technology. This means that the physical and data link layer are not specified and so any (existing) physical/data link layer combination can be used. For reasonable communication, only an unconfirmed unicast and a broadcast communication service are considered mandatory. Still, it is the goal to reuse existing services rather than define new ones.

In addition to native data link layer protocols, it shall also be possible to use higher layers (i.e., layer 3 and above) as data link layers. To achieve this, a so called *Virtual Data Link Layer (VDLL)* has to be included. This VDLL provides an interface for the SAL and thus simulates the use of the underlying protocol layers as native data link layer. A typical example would be the use of IP as data link layer for the SAL. This concept is similar to BACnet/IP where UDP is used as data link layer [7].

3.2. SAL

The main aim of the SAL is to use the communication services of the NSL and provide generic secure communication services to the ASL. The SAL corresponds to the OSI layers 3 to 6. To support different kinds of ASL application layer models, it provides different types of generic communication services. The objectives of the SAL are:

- *Communication service types:* In BAS, both management and process data exchange are common. Since their characteristics vary (e.g., sporadic management tasks vs. frequent process data exchange), also the demands on the communication service differ. Therefore, (multiple) services that meet the demands of both types of data exchange have to be provided by the SAL.
- *Communication models:* Depending on the application layer models used, support for different communication models is required. For example, BACnet's client/server model demands only a unicast service, while KNX's producer/consumer model relies on multicast. Finally, for management tasks most application layer models use broadcasts. In any case, it is the task of the SAL to simulate a service if the underlying layer does not natively support it (e.g., emulate multicast by using multiple unicasts or a broadcast).
- *Global naming:* For routing across heterogeneous network segments, a global naming scheme based on global *SAL addresses* is mandatory. Using the SAL addresses, the different addressing schemes of the data link and application layer models can be abstracted. The SAL is capable of mapping the global addresses to the corresponding data link addresses as well as to the application service access points (ASAPs) of the application layer.

Between the SAL and the NSL so called *data link plugins* are located. These plugins provide an abstraction of the underlying data link communication services. Each data link plugin is dedicated to a specific data link/physical layer combination. Devices that have more than one interface to heterogeneous networks thus also need to implement one data link plugin for each network.

The main objective of a data link plugin is to select the data link communication services that are suited best to fulfill the above mentioned objectives of the SAL. Furthermore, it is geared towards sensible (re-)use of already existing data link primitives. This means that each plugin chooses the services that fit best for the requested communication service. Consider, for example, the data link layer of IEEE 802.15.4, which provides security at the data link layer. The corresponding data link plugin can make use of these services in case that they meet all requirements imposed by the request. Another example is the reuse of the existing multicast communication services of KNX and LonWorks once the ASL requests to send a message to multiple receivers. In contrast to that, the use of unsuited protocol features (e.g., LonWorks security is considered insecure [8]) can be blocked by the plugin.

The SAL is supported by different *helper modules* to be able to fulfill all communication objectives: The *naming/routing module* is responsible for the management and

translation of the SAL addresses. This includes a translation of the data link layer addresses to the SAL addresses with the help of an address translation table. Furthermore, it implements different routing services that are used, e.g., for routing table maintenance.

The *cryptographic module* encapsulates different cryptographic algorithms and primitives that are used to guarantee the required security objectives that cannot be fulfilled by the NSL (e.g., strong encryption). This mainly includes data integrity, freshness and confidentiality. However, since the available device resources of embedded devices (i.e., processing power and available memory) are limited, only those security objectives that are required by the desired application have to be guaranteed by the SAL ("good enough security"). Furthermore, the module is responsible for the management of shared secrets that are used as input parameters for the cryptographic algorithms (e.g., shared secret keys, nonces), including management services to retrieve and revoke these shared secrets.

The communication services provided by the SAL are accessible through a generic communication service interface. It is used by so called *application layer plugins* that in turn interface with the ASL.

3.3. ASL

The ASL corresponds to the application layer of the OSI reference model. It builds upon the communication services provided by the underlying SAL and offers an API to user application(s). As for the NSL, the proposed architecture neither specifies a new application layer protocol nor it is limited to any kind of user application.

On the one hand, this has the advantage that any application layer protocol can be used. If required, any protocol can be implemented by an application layer plugin that is located on top of the SAL. On the other hand, the ASL allows a device to host multiple user applications, even of different technologies. This support for standard user applications of different technologies demands that multiple user layer plugins are included in the device. These plugins use the generic services of the ASL and provide a technology-specific interface to the user application.

The ASL itself is responsible for management of the device's data points. All data points of a device are represented as so called application objects (AOs) and stored in a generic *application object database*. This way, the objects are accessible by all user and application layer plugins and can be manipulated in various ways. For example, an application object can be accessed by a BACnet user application while interfacing with a ZigBee application layer plugin. Configuration and maintenance e.g., creation and removal of AOs, shall be possible with management tools (ideally already existing tools) interfacing the database via management plugins.

Obviously, the structure and semantic of the stored data values (e.g., encoding) as well as the methods used to access the data points (e.g., addressing, binding) can be different for each technology employed. Therefore, some kind of data point mapping and translation is necessary. Tool support for a comprehensive definition of mapping and translation rules is still work-in-progress, with the ultimate goal to accomplish the task semi-automatically with the help of (probably extended) management tools.

4. Conclusion and Work-in-Progress

The main feature of the proposed solution is flexibility. Due to the plugin-based design of the communication stack, the presented architecture is not bound to a specific technology. This concerns the used application layer and network medium as well as the user application(s). Also, the new devices working with the multi-protocol stack are fully compatible to be used in existing installations and can coexist with regular devices. Finally, the security abstraction layer enables end-to-end security among any group of devices.

As a first proof-of-concept, a multi-protocol device that supports KNX and BACnet applications has been developed [9]. The device is based on a MSP430 and uses KNX TP 1 as network medium. On top of the application layer protocols, a generic application object database was implemented and a BACnet/KNX data point mapping was defined.

As next steps, plugins shall be added that support wireless and other network media and the data point translation rules shall be refined. A starting point for the latter is the data point abstraction layer model of [10]. Additionally, options how the modular SAL can be updated or enhanced at runtime will be evaluated. Finally, also the management of the mapping shall be facilitated by providing management clients access to the AO database via plugins.

References

[1] S. Soucek and D. Loy, "Vertical Integration in Building Automation Systems", in *Proc. 5th IEEE Int. Conf. on Industrial Informatics*, June 2007, pp. 81–86.

[2] ZigBee Alliance, "ZigBee Specification", 2006.

[3] "BSR/ASHRAE Add. g to ANSI/ASHRAE Standard 135-2004", April 2007, Second Public Review completed.

[4] "KNX Specification", Version 1.1, 2004.

[5] "Control Network Protocol Specification", ANSI/EIA/CEA 709.1, 1999.

[6] IEEE Computer Society, *IEEE Std. 802.15.4-2006*, 2006.

[7] "BACnet – A Data Communication Protocol for Building Automation and Control Networks", ANSI/ASHRAE 135, 2004.

[8] W. Granzer, W. Kastner, G. Neugschwandtner, and F. Praus, "Security in Networked Building Automation Systems", in *Proc. 6th IEEE Int. Workshop on Factory Communication Systems* , June 2006, pp. 283–292.

[9] W. Granzer and W. Kastner, "BACnet over KNX", in *Konnex Scientifc Conference*, November 2007.

[10] W. Burgstaller, S. Soucek, and P. Palensky, "Current Challenges in Abstraction Data Points", in *IFAC Int. Conf. on Fieldbus Systems and their Applications*, 2005, pp. 40–47.