# Security requirements of IoT-based smart buildings using RESTful Web Services

**Marco Niemeyer, Kilian Henneböhle, Markus Kuller, Ingo Kunold**

[Institute of Communications Technology, Dortmund University of Applied Sciences and Arts, Sonnenstr. 96, 44139 Dortmund, Germany], ikt(at)fh-dortmund.de

*Since the introduction of modern web technologies into the traditional market of home automation systems it is the idea to achieve the full interconnection not only on the physical layer but also on the application layer. The vision of smart devices which are interacting and communicating with each other is the so called Internet of Things (IoT) and it seems its realisation takes place in the near future. Recently it was hardly possible designing a smart building without using at least one or more proprietary technologies and software stacks which resulted in a time-consuming and cost-intensive realisation process.*

*By the implementation of a software wrapper for each technology and the provision of adequate APIs it is now quite comfortable to create an IoT middleware which covers all features which are essential for a smart building.*

*By using open standards and connecting all devices across the internet it is necessary to carefully take all security aspects and risks into account. Certain guidelines and requirements are already provided by national authorities. The technical guideline [1] and the Protection Profile [2] cover cipher suites, encryption methods and recommendations for the architecture of the network with the focus on Transport Layer Security (TLS).*

*This paper deals with the security requirements of the IoT which is an essential part of a smart building. The important web technologies are introduced and compared.*

*One solution is the use of RESTful Web Services to achieve the goal of a fast, efficient and secure software architecture. As an alternative technology the WebSocket protocol is also presented in this paper.*

*Keywords: Internet of Things, RESTful, Web Service, WebSocket, smart building, Transport Layer Security, cipher suites, middleware*

## 1 Introduction

The Internet of Things is the evolution of devices which are smart and get interconnected to the internet. For this purpose devices become intelligent, e.g. in form of sensors and actuators with an integrated microcontroller and a communication interface. Now they are ready to act autonomically, communicate with each other or getting controlled by external entities. Human interaction is not necessary anymore, machines are interacting with machines (M2M communication). The IoT devices enable new appliances in the field of *smart buildings*. Any sensors and actuators can be used to receive real time status information or control the building infrastructure. The processes are called *smart building services*. These services can be deployed on an embedded hardware platform.

A big problem is the missing of a uniform standardization on the physical and the protocol layer of the IoT fieldbus. A possible approach for standardization on the protocol layer is an actual field of research.

With this drawback in mind it is a main goal to introduce a standardized data model that describes every essential function in a smart environment.

Usually a developer needs in-depth knowledge of the specification of the fieldbus technology to be implemented. By using a standardized data model it is possible to use an API that covers IoT functions independently of the physical layer below. In this case protocols for the session and data transport have to be applied. It is a popular approach nowadays to use modern and established web technologies for the IoT – on the contrary to the ancestors of M2M communication using proprietary technologies – on the purpose to simplify the development. This modern approach offers both opportunities and risks, a web architecture is highly vulnerable to attacks from the internet, therefore appropriate measures have to be taken.

In this paper we focus on the RESTful Web Service and alternatively we give an exemplary concept of a scenario with WebSocket technology.

## 2 Related Work

The research project INES [3] deals with a local optimization of load- and generator-based capacities on the basis of the individual user profile and predictive load-analysis.

This concept was developed as part of the research in the Dortmund Institute of Communications Technology. It will be deployed within the Federal Ministry for

Economic Affairs and Energy (BMWi) research project GUIDED AB [4] which is joint by several consortium partners of industry and research.

## 3 Security objectives and technical requirements

In accordance to the security in a smart grid/ smart metering environment the Federal Office for Information Security (BSI) has defined the Protection Profile (PP) [2] along with the technical guideline TR-03109 [1] which define the requirements for a Smart Meter Gateway (SMGW) and its interaction with other components in a *smart building*.

The IoT devices in this work shall be implemented to fulfill certain security requirements of the BSI PP. Generally every communication channel must be secured by TLS. This includes the data traffic between the IoT Interface and authorized external entities, e.g. the IoT server or the Smart Building Services. It is mandatory that the connection to the Wide Area Network (WAN) has to be established by the Smart Building Manager (SBM) as described in the following chapter [5]. For connection calls from outside wake-up procedures shall be provided as required in the PP.

## 4 Middleware concept and network architecture

The field of research of the Dortmund Institute of Communications Technology [6] [7] is the integration of IoT middleware concepts into a sustainable building network technology. Prototypical solutions of Human-to-Machine (H2M) and Machine-to-Machine (M2M) interactions are developed in three steps. This cycle covers concept, design and a complete platform including hard- and software.

Figure 1 shows the INES IoT system architecture with all its instances. The IoT Server and Smart Building Services such as home and building automation services or the prediction of power consumption may be located in the cloud. The IoT Server as a backend system has the task of long term storage of condition and metering values from the IoT actuator and sensor network. Additionally it maintains a role based and user access control list (ACL) to allow read and write access to the devices. To realise this access an IoT interface located between the middleware and the cloud services will be defined. On the middleware level the distributed architecture has to run nearly under real time conditions. The middleware concept includes a SBM and a Smart Device Controller (SDC) as depicted in Figure 1. In addition to the addressing function the SBM is responsible for providing near real time status information of the building over RESTful Web

Service. One of the functions of the SDC is the mapping of a proprietary to a standardized data structure. At the level of the fieldbus and wireless network different protocols will be applied.

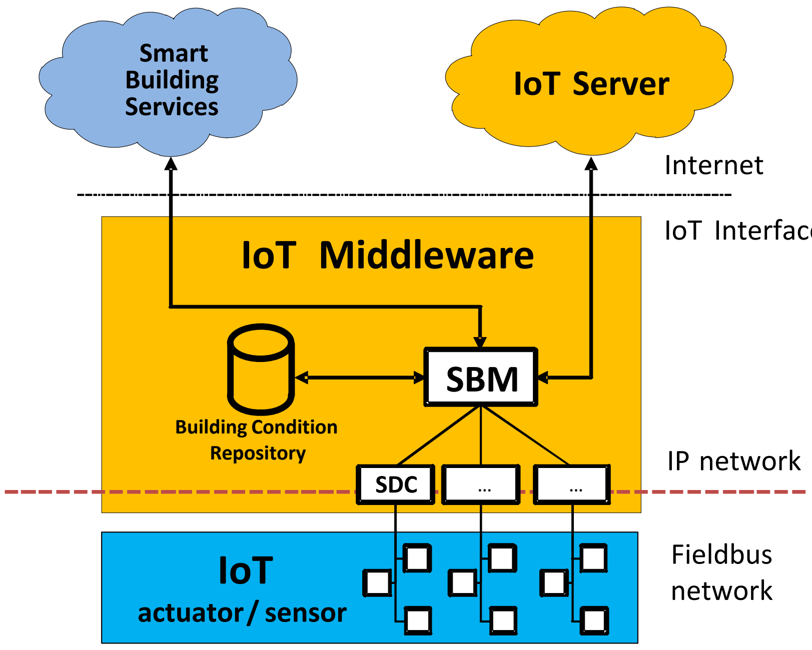

Figure 1
INES IoT System architecture

The IoT interface must meet different security requirements. Taking into account that sensitive private data is handled, the communication channel has to be secured by proven mechanisms like SSL/TLS.

In this context the level of encryption depends on the applied cipher suites. Guidelines can be used which are provided by national authorities as already mentioned before. Furthermore the IoT interface must support a user and role based authentication to manage the access to its resources characterised by the REST architecture. To offer the highest security level only authorised instances communicate with each other. This requires both instances performing a mutual authentication before an encrypted data transmission is initiated. This ensures a very high grade of interception and manipulation security.

## 5 IoT Interface

The INES realisation of the IoT interface is a Web Service in REST architecture. Other Services (e.g. IoT Server or Smart Building Services) can access the Web Service through resources. These resources provide building condition values or metering data in variable granularity from the sensor and actuator network. In addition the resources provide building control functions which are excecuted in the fieldbus network.

To identify these resources URL addresses are used. The URL address describes the path from the SBM via the SDC through the sensor and actuator network. Every instance in the IoT middleware concept can be identified by a Universally Unique Identifier (UUID) that is also a part of a URL. As a result a URL in the IoT middleware can be represented as follows:

*https://<WebServerName>:<Port>/SBM/<uuid>/SBC/<uuid>/Sensor/<uuid>*

If a resource is requested by an external service the response of the Web Service is represented in the INES data model called *DeviceData* formatted in Extensible Markup Language (XML) or JavaScript Object Notation (JSON) according to the usual practise in the web. Caused by the interconnection of different units in a building the IoT interface has the functional requirement to ensure data security as well as the managing of access rights. The last point is important if the IoT system performs crossover-processes beyond the building units.

## 6 RESTful Web Service

The REpresentational State Transfer (REST) is a common architectural style widely used for web-based M2M communication and thus well suited for the IoT [8].

The technology is the so called RESTful Web Service and it is nowadays a popular and more light weight alternative to the well known SOAP Web Service. RESTful Web Service uses the stateless communication protocol HTTP. The concept is to offer resources over a web server which are uniquely identified and accessed over the standardized Uniform Resource Identifiers (URI). Resources are the essential part of the concept, they can be manipulated using the HTTP methods PUT, GET, POST and DELETE as defined in RFC 2616 for the four CRUD (Create/Read/Update/Delete) operations.

Another advantage is the use of important features of the HTTP protocol. For example it is possible to use resource-dependent HTTP Caching to decrease network load and latency.

Resources can be provided in different formats and there are different methods of accessing them. This is called the representation of a resource. Depending on the client and the request parameters data can be provided in a variety of standardized formats, e.g. as plain HTML, JSON or XML. The metadata contains information about access control and authentication.

On the contrary to SOAP-based Web Service the REST architecture has the advantage of less overhead because of the shorter requests and responses.

# 7 Security aspects

With the introduction of the Internet of Things the security paradigm - compared to conventional IT systems – has changed. A transition from closed networks to the public internet takes place. Regarding this fact the protection of personal data becomes an important issue. In an IoT system many devices like sensors or actuators are located and each of them has to fulfil stringent security critera. The networks themselves have also be secured with additional security measures to mimimize weak points of potential attacks. This includes the data protection on the transport layer of the IoT system. The information to be transmitted will be encrypted to protect the confidentiality and integrity. The use of certificates in a public key infracstructure ensures the authenticity. Relating to this the highest level of security will be given through mutual (server/client) authentication. All of this is applied in the settings of the cipher suites in the TLS protocol. After the handshake between client and server the cipher suite list is exchanged. In this list different security parameters like key exchange or encryption algorithms are defined. In the IoT system the Security-API Java Secure Socket Extension (JSSE) is used. This API contains an implementation of the TLS protocol and especially in the so called Cryptographic Service Provider, a list of supported cipher suites as defined in Java. In conformity of the TR-02102-2 [9] and the recommendation of the Internet Engineering Taskfore (IETF) on the safe use of TLS [10] following cipher suites will be applied :

- TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256
- TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384

As previously mentioned a mutual authentication between client and server is handled. In the IoT system – based on a public key infrastructure – every communication instance has exclusive certificates that control the permission to access to other instances. This means that only instances can build-up a transmission channel if they trust each other. This prevents man-in-the-middle attacks against the IoT system.

Access to a Web Service resource assumes an authentication via login name and password. At this point the HTTP Authentication Scheme is used so that credentials are not transmitted in clear text. Furthermore every request to a resource requires a session dependent token that is provided after a succesfull login to the IoT system. The randomly generated token loses its validity after an expiration time or the termination of the session.

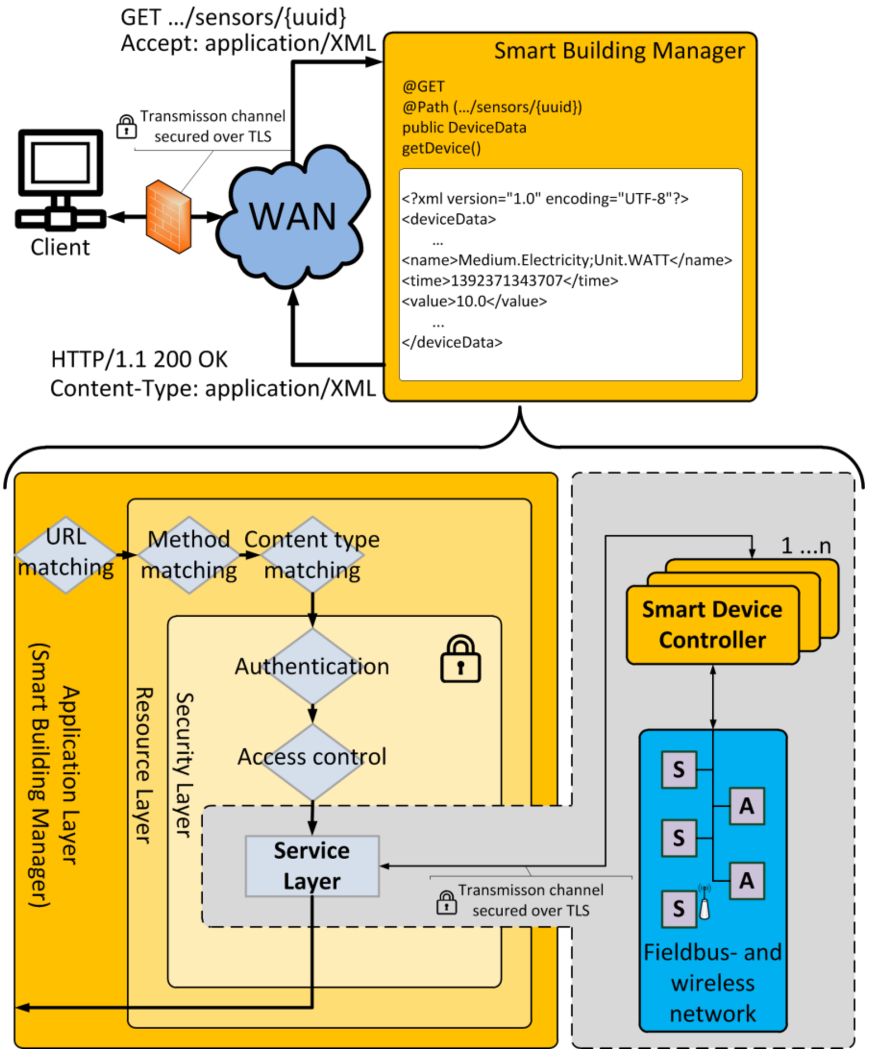

Figure 2
REST Security mechanism

Figure 2 shows an exemplarily Web Service call from a client to the IoT interface to request sensor data with a defined UUID. First of all – after the HTTP request is received by the SBM – the URL of the requested resource is checked for the existance on the Application Layer. After a successful check a verification of the HTTP parameters on the Resource Layer (method and content type) will be done. The following two steps including authentication and access control are assigned to the Security Layer. The request is processed on the Service Layer by the SBM using the actual sensor data of the Building Condition Repository. The Respository is updated by the internal communication chain between SBM – via the SDC – and the sensor on the fieldbus layer.

## 8 WebSocket

Originally the WebSocket protocol was designed for HTML5 & JavaScript browser applications to avoid the polling problem of many web applications. Generally it enables applications to open a persistent connection to the server and communicate in near real time and low latency therefore it is well suited for the M2M communication in the IoT. The communication channel is bi-directional and full duplex over a single TCP connection. On the contrary to HTTP, WebSocket is a protocol with no pre-defined message pattern like request/response. Either the client or the server can send data to each other in an event based messaging style. The communication can take place in both directions independently. After the HTTP handshake the connection is upgraded to a TCP-based WebSocket connection and is maintained for the whole lifecycle. The HTTP Upgrade is a standard mechanism and can be interpreted by any HTTP servers. Another advantage is the lacking of the HTTP overhead, thus offering more bandwidth. Data is sent in binary form or UTF-8 encoded.

The client can be implemented in various ways as the WebSocket library is provided for many programming languages.

The communication is done over TCP port 80 for HTTP therefore most firewall configurations will allow WebSocket communication passing through. On the other side, blocking of the communication is not possible if the firewall only works on OSI layer 3 or 4 and has no further information about the protocol.

WebSocket is still missing important features which are actually provided by RESTful HTTP. Procedures like creating, updating or deleting resources have yet to be implemented for WebSocket. The protocol was standardized by the IETF as RFC 6455 [11].

## 9 Conclusion

Any physical process in a *smart building* can be interpreted as a synchronous process with its specific sampling rate or as a stochastic process with static or dynamic threshold values which generate trigger system events.

In the INES project the related process data are encrypted and transferred to the Building Condition Repository of the SBM. The data access on the Internet or Intranet site is realised by the IoT Interface using the descripted methods RESTful Web Services and WebSocket.

The suitable access and security mechanisms for the IoT is recently a topic in several working groups for standardization on a national and international level, e.g. [12] [13].

## 10 Acknowledgement

This work has been realised within the research project INES funded by the Federal Ministry of Education and Research (BMBF).

**References**

[1] BSI - Federal Office for Information Security, “Technische Richtlinie BSI TR-03109,” March 2013. [Online]. Available: https://www.bsi.bund.de

[2] BSI - Federal Office for Information Security, “Protection Profile for the Gateway of a Smart Metering System (SMGWPP),” March 2013. [Online]. Available: https://www.bsi.bund.de

[3] K. Henneboehle, N. Karaoglan, M. Kuller, I. Kunold, “Decentralized Energy Equalizer for a balancing aggregation of production and consumption of energy in scalable units,” in *IEEE IDAACS’2013 (Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications)*. Berlin: IEEE, September 2013.

[4] BMWi - Federal Ministry of Economics and Technology, “GUDED AB,” 2013-2016. [Online]. Available: http://www.guided-ab.de/

[5] M. Kuller, I. Kunold, H. Hoffmann, *Middleware- und Visualisierungskonzepte für Smart-Energy-Systeme*. vwh | Verlag Werner Hülsbusch, 978-3-86488-055-1, November 2013, pp. 42 – 55.

[6] “Dortmund Institute of Communications Technology.” [Online]. Available: http://www.fh-dortmund.de/ikt

[7] K. Henneböhle, M. Kuller, I. Kunold, *Verteilte Architektur für eine ausgleichende Aggregation von Verbrauch und Erzeugung von Energie in Privathaushalten*. Verlag Werner Hülsbuch, ISBN: 978-3-86488-030-8, November 2012, pp. 123 – 131.

[8] Roy Thomas Fielding, "Architectural Styles and the Design of Network-based Software Architectures," Ph.D. dissertation, University of California, Irvine, 2000.

[9] BSI - Federal Office for Information Security, "BSI TR-02102-2 Verwendung von Transport Layer Security (TLS)," February 2014.

[10] IETF, *RFC 5246 - The Transport Layer Security (TLS) Protocol Version 1.2*, IETF Std., August 2008. [Online]. Available: http://tools.ietf.org/html/-rfc5246

[11] IETF, *RFC 6455 - The WebSocket Protocol*, IETF Std., December 2011.

[12] DKE - German Commision for Electrical, Electronic and Information Technologies of DIN and VDE, "DKE/AK 716.0.1." [Online]. Available: http://www.vde.com/en/dke/Pages/DKE.aspx

[13] FNN - Forum network technology/ network operation in the VDE. [Online]. Available: http://www.vde.com/en/fnn/Pages/default.aspx