УДК 519.718.7:681.3.069

Surkov S. S.,
Martynyuk O. N., PhD

COMMUNICATION MODEL AND PROTOCOL BETWEEN SMART HOME COMPONENTS

Abstract. Internet of things (IoT) and smart home play more and more important role in people's life. The foundation of our smart home solution is communication protocol which needs to be reliable and protected against network attacks. This paper proposes the communication model between smart home components such as and non-platform dependent central server and smart home modules which is based on WiFi microcontrollers.

Keywords: *microcontrollers, protocol buffers, smart home, binary protocol, websockets*

Сурков С. С.
Мартынюк А. Н., канд. техн. наук

МОДЕЛЬ И ПРОТОКОЛ КОММУНИКАЦИИ КОМПОНЕНТОВ УМНОГО ДОМА

Аннотация. Интернет вещей (IOT) и умный дом все больше играют важную роль в жизни людей. Основой исследования и разработки является протокол, который необходим быть надежным и защищенным от сетевых атак. Данная работа предлагает модель коммуникации между компонентами умного дома, такими как платформо-независимый центральный сервер и умный модуль, который базируется на WiFi микроконтроллере.

Ключевые слова: *микроконтроллеры, protocol buffers, умный дом, бинарный протокол, websockets*

Сурков С. С.
Мартинюк О. М., канд. техн. наук

МОДЕЛЬ І ПРОТОКОЛ КОМУНІКАЦІЇ КОМПОНЕНТІВ РОЗУМНОГО БУДИНКУ

Анотація. Інтернет речей (IOT) і розумний будинок все більше відіграють важливу роль в житті людей. Основою дослідження і розробки є протокол, котрий необхідний бути надійним і захищеним від мережевих атак. Дана робота пропонує модель комунікації між компонентами розумного будинку, такими як платформо незалежний центральний сервер і розумний модуль, який базується на WiFi мікроконтролеру.

Ключові слова: *мікроконтролери, protocol buffers, розумний будинок, бінарний протокол, websockets*

Introduction

Smart Home is a hardware and software solution to establish control of smart home devices by PC/Smartphones over WiFi. The foundation of our Smart Home solution is communication protocol between central server [1,2] and smart home modules and control device. This work proposes communication protocol structure between smart home modules, in which important part plays Google ProtoBuff [3]. Using this protocol allows to serialize complex data structures in binary format without the necessity of developing custom serialization protocol.

Aim of the work

Aim of our work is to research and develop communication model for Smart Home, which would allow make fully automated home using Smart Home with:

- easy integration of Smart Home Modules to existing networks [1, 2];
- control smart home from anywhere via Internet;
- achieve high reliability and security for Smart Home Network;
- Central Server run on any platform.

Our approach can be applied to: turn on/off lights, power sockets, etc, control PC power, dim lights, smart door locks (using Touch IDs), smart sprinkles, etc.

We plan to apply our approach for:

- Have control over all the lights and electric devices which needs to be turned on/off.
- Dim light in bedroom.
- Observe PC state and be able to press

Power/Reset on your home PC remotely.

- Have temperature sensors outside the window.
- Control IR devices such as TV, Speakers, Air Conditioner, HDMI switch, etc...
- Turn on/off devices remotely in dangerous environments.
- Read data from various of sensors.
- Observe Servers state and be able to press Power/Reset on servers remotely.

Analysis of existing communication protocols

To choose communication protocol there were investigated three common protocols, which are used for various of Smart Home implementations: WiFi, ZigBee, Develop custom radio protocol.

The main advantage of ZigBee protocol is low power consumption, but at the moment of publication of this article we didn't find low-cost MCUs, which would support this protocol. In addition, for ZigBee networks it would be required to buy additional hotspots, which would require Wi-Fi/Lan – ZigBee bridges. The development of custom radio protocol would require much efforts to develop hardware and software part.

To simplify the development of communication model we decided to use WiFi protocol, which would not require to modify additional network infrastructure, with very cheap and powerful WiFi ESP8266 MCUs as a base for Smart Home Module, Cactus Micro (Arduino with ESP8266 on board).

Analysis of hardware and software platforms

As a central server we will make it a software solution without any hardware/platform dependencies. To run on many platforms as possible we plan to use java. For standalone central server solution, it makes sense to run the server on dedicated PC/Development Kit (e.g. Raspberry Pi) with low power consumption. It also doesn't matter, if the central server will run on a server which serves for other purposes plus smart home server or on standard. The only requirement for central server is to work at the time, when all the smart home modules do, which in most of the cases means to work 24/7. The choice of central server platform depends on requirements and available resources of person or organization.

Modules which we use are shown in figure 1.

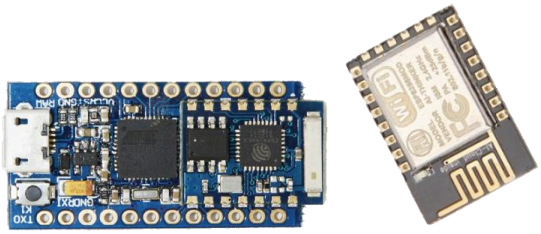

Fig. 1 - Cactus Micro and Esp 8266 (ESP12-E) boards

As use case for user control device needs to have access to central server and may be absolutely any platform. After successful connection authorized device from central server gets list of modules and allows to controls them.

General structure of Smart Home Network is shown in figure 2.

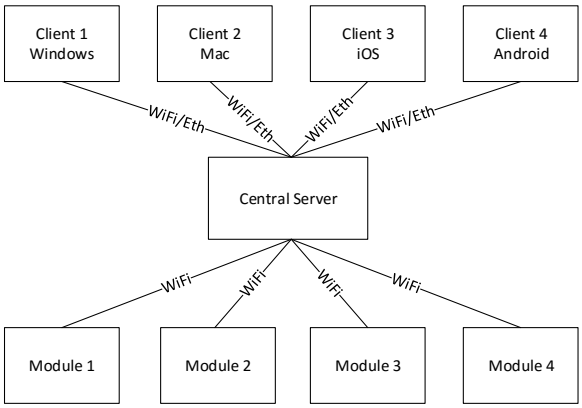

Fig 2: General structure of Smart Home Network

Central server is developed in Java with Websockets library and default protobuf code generator. Modules connect to central Node (Raspberry Pi), when they're started. To develop application for a platform there's only requirement to have websockets [4] library and protobuf [3] support. For ESP8266 nanopb and nopoll libraries were used. To secure all the communications we plan to use and modify approach from our researches [5, 6, 7] to work with binary data.

Protocol Structure

To communicate between components, Web Socket [4] protocol over TCP is used. Other than this, a command may be transferred via unreliable UART interface between AtMega and ESP8266 chips. To prevent data corruption during transmission by UART port CRC16 fields were added. Protocol frame structure is shown in figure 1:

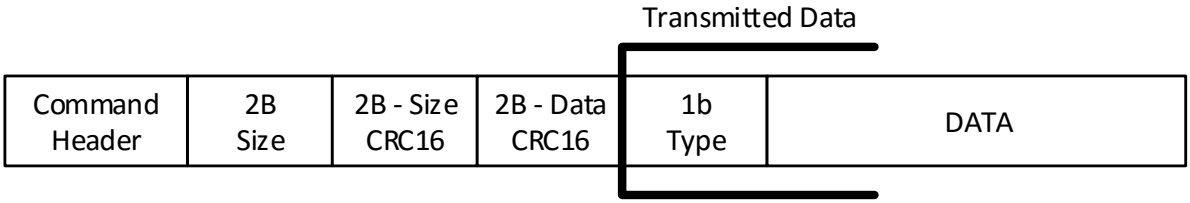

Fig. 3 Smart Home Protocol Structure for Serial and WiFi connection

The section in transmitted data is used for actual data and type of data. The other part of the message is used to verify the integrity of the data in case of unreliable communication channel. "Command header" field is unique header, which purpose is to determine, if the new command is started. "Size" field is used to detect how many bytes is expected to retrieve. "Size CRC16" and "Data CRC16" are used to check consistency of transmitted data and size field. If CRC16 of size field or transmitted data doesn't match the calculated CRC16, it means, that data is corrupted, and in this case command will be skipped.

For websocket communication only transmitted data section is sent through websocket frame. However entire structure may be transmitted via websocket in the case, if the structure is going to be sent via UART interface. This way we decrease the load for a module and simplify message handling logic on the ESP8266 MCU.

In transmitted data section the first byte is type of Protobuf message, which is going to be deserialized. To simplify development and ensure consistency of the data we suggest to define the values for it right in .proto file. After matching the data type, data is deserialized through protobuf library.

Communication Model

In communication model there are three components: client, server and module. All the communication goes from client to server and then server sends request to module. The sequence diagram is shown in figure 4:

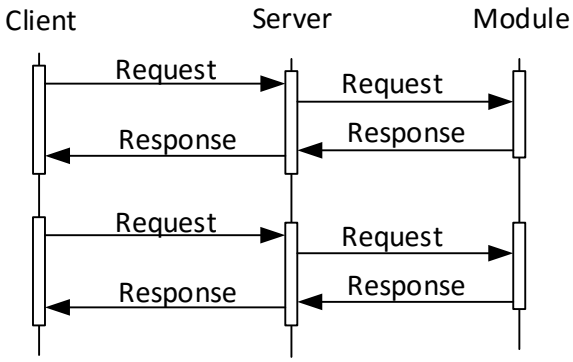

Fig 4. Sequence diagram of communication

In the case of single client, the only advantage of WebSocket technology over HTTP may be amount of received and transmitted data. This statement will be correct, if the total size of ping/pong frames, which are used to keep the connection alive will be less amount of HTTP headers for all of the requests.

The real advantage of websockets protocol will be in the case, if there're several clients are connected to the central server. The sequence diagram for case with multiple clients [8] is shown in figure 5.

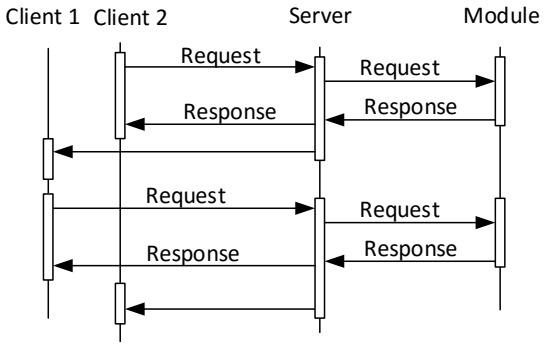

Fig 4. Sequence diagram of communication for multiple clients

In this case server simply notifies other client about change of state for module. It gives to a client application instant updates of the module state. Other than this it gives significant reduce for data, which needs to be transferred. In case of websockets neither continuous status update requests neither long pull requests are required.

To define communication between components it's important to create high-level definition of the protocol. The UML diagram of communication protocol is shown in figure 5:

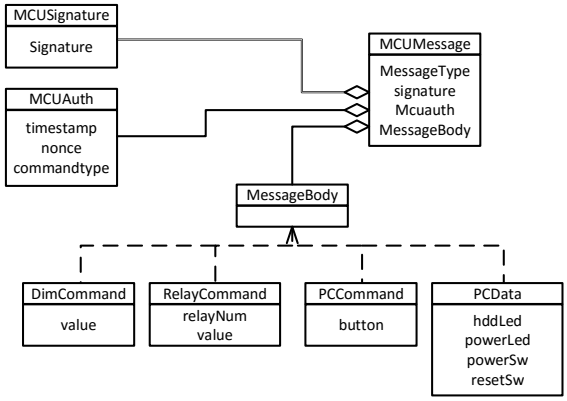

Fig 5. UML diagram of Communication protocol

There are three main entities on the diagram: MCU Message, MCU Signature, MCU Auth. MCU Message entity is base entity which is used for communication between components and contains 4 fields: Signature, Auth, Message Type, Message Body. The purpose of Signature and Auth fields are for digital signature of the message in order to prevent any modification of the protocol during transmission. Message type field is used to find what message type is wrapped. And Message body is actual message which is going to be handled by Client/Central Server/MCU.

Security of the protocol

Commands at the moment of transmission through network are vulnerable to any kind of analysis and attacks. The other case might be at the moment of establishing connection to server [5], if rogue server will be found in network. To secure messages HMAC approach is used.

To ensure that the same message [6, 7, 8] won't be sent twice Nonce and Timestamp are used. Nonce is pseudo-random generated number which is used to be sure the request wasn't sent before and timestamp is used to ensure that request is neither too old or too new. Checking timestamp allows to keep nonces only within current time. Of course for MCUs there's no possibility to keep large amount of data and to verify nonces cyclic array is used. By checking nonces and timestamps we prevent reusing already sent messages. This protocol of authorization we call ProtoAuth. Structure of ProtoAuth message is shown in figure 2.

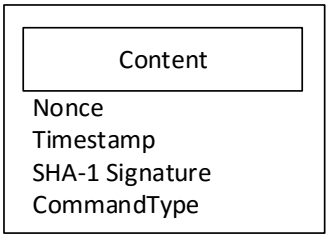

Fig 4. Structure of Websocket Message

The main field in websocket message is signature field, which purpose is to verify, that message is came from trusted server and wasn't modified during transmission. The SHA-1 signature is taken from other part of message plus shared secret. Shared secret is the key private parameter in this signing the message, if it's exposed to attacker then he can easily hack the system.

For HMAC systems it's proven, that SHA-1 algorithm is very secure. Finding of SHA-1 collisions in HMAC case may make a system unstable with high cost of finding them, only finding original message may endanger the system. Other than this with today computing powers finding a collision costs about $75,000 and $120,000 using computing power from Amazon's EC2 cloud over a period of a few months [9]. These kind of attacks are used to break HTTPS SSL certificates, but for HMAC case with message 30 minutes' live time now it's no

threat at all.

Conclusion.

In this work we performed analysis of the existing communication protocols to build Smart Home and we found, that to build Smart Home, which uses WiFi as to communicate, will be the most cost effective and requires the least efforts to implement approach. This approach allows to avoid the development hardware solution or requirement to buy additional for central server as existing one could be used.

The newly developed communication model allows to send and receive updates from central server by multiple clients and work with several modules simultaneously.

Other than this we developed communication protocol, which reliable serial communication and secure communication through WiFi. For serial communication we developed protocol structure of Smart Home message, which allows verifying integrity of the command. For WiFi communication we developed a secure protocol based on HMAC, which makes almost impossible to modify the data during transmission without knowing shared secrets. The result of our research might be used in many areas such as Smart Home, embedded systems, REST API development and gives to the systems additional security and reliability.

Received 03.05.2016

References

1. S.S. Surkov, O.M. Martynyuk Research & Development of Smart Home Protocol structure, safety and security, Proceedings of the 2016 MIET Conference

2. S.S. Surkov, O.M. Martynyuk Research and development of smart home communication model, Proceedings of the 2016 MIT Conference

3. Google Protocol Buffers https://developers.google.com/protocol-buffers/

4. The WebSocket Protocol. Internet Engineering Task Force (IETF) I. Fette, Google, Inc., A. Melnikov ISSN: 2070-1721 Isode Ltd., December 2011, https://tools.ietf.org/html/rfc6455

5. S.S. Surkov, O.M. Martynyuk, I.G. Mileyko Modification of open authorization protocol for verification of any HTTP request. – Electrotechnic and Computer Systems, Special Edition 2015

6. S.S. Surkov, O.M. Martynyuk Method of Migration from Single Server System to Server Cluster. Proceedings of the 2015 IEEE 8th International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS'2015)

7. S.S. Surkov, A.N. Martynyuk Authorization for automobile headunit without browser support with mobile devices through Bluetooth - Refrigeration engineering and technology №2, 2015

8. Foss S., Korshunov D. Heavy Tails in Multi- Server Queue // Queueing Systems. 2006. Vol. 52.

9. New Collision Attack Lowers Cost of Breaking SHA1 http://www.securityweek.com/new-collision-attack-lowers-cost-breaking-sha1

Сурков
Сергей Сергеевич;
аспирант кафедры Компьютерных интеллектуальных систем и сетей Одесского национального политехнического университета
тел.: +38(091) 916-40-91
E-mail:
k1x0r@ukr.net

Мартынюк
Александр Николаевич,
канд. техн. наук, доцент каф. Компьютерных интеллектуальных систем и сетей Одесского нац. политехн. ун-та.
тел.: +38(067) 489-81-69
E-mail:
anmartynyuk@ukr.net