Object Secured TCP Socket for Remote Monitoring IoT Devices

Rohan A. Nathi
S. N. Systems Pvt. Ltd, Pune, India
rohan.nathi@gmail.com

Dimpal Sutar
Zhypility Technologies Pvt Ltd,
Mumbai, India
dimpalssutar@gmail.com

Abstract—With evolution of the communication technology remote monitoring has rooted into many applications. Swift innovation in Internet of Things (IoT) technology led to development of electronics embedded devices capable of sensing into the remote location and transferring the data through internet across the globe. Such devices transfers the sensitive data, which are susceptible to security attacks by the intruder and network hacker. Paper studies the existing security solutions and limitations for IoT environment and provides a pragmatic lightweight security scheme on Transmission Control Protocol (TCP) network for Remote Monitoring System devices over internet. This security scheme will aid Original Equipment Manufacturer (OEM) developing massive IoT products for remote monitoring. Real time evaluation of this scheme has been analyzed.

Keywords—-Internet of Things(IoT), Secure Socket Layer(SSL), Transmission Control Protocol (TCP) , Global System for Mobile communication(GSM),General Packet Radio Service (GPRS) , Remote Monitoring System(RMS);

I. INTRODUCTION

Remote Monitoring System (RMS) has firmly established in industrial, healthcare, agricultural and meteorological applications. Technological Advancements in IoT has led to revolution in Remote monitoring system, by enabling monitoring across the globe. Figure 1 shows RMS IoT network architecture. Authors in work [1, 2] has presented hardware and software architectures for RMS IoT devices. Figure 2 present the generic remote monitoring device architecture for IoT in our work. This architecture gives and brief overview of software and hardware aspect of the RMS devices. The architecture is divided into 3 layers and described as follows.

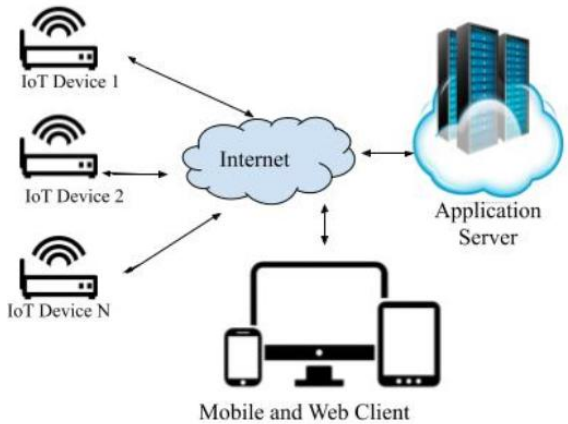

Figure 1:RMS Network Architecture

1) *Sensing Layer:* Sensing layer consist of sensor network responsible for sensing the raw data. It can consist of variety of sensor including temperature, humidity, moisture etc. depending upon the application. Signal Conditioning is performed which includes amplification, conditioning, filtering and A/D conversion techniques.

2) Presentation Layer: This layer is responsible for presentation of the collected data from the sensor network. JSON and XML formats can be used to represent data. For example JSON format that represents the data from sensor network is shown {“sensor1”:12.6, “sensor2”:700, “sensor3”:123}.

3) Network Layer: Network layer is responsible for transferring the data across the internet. Wi-Fi Modules GSM/GPRS Modules and Ethernet Modules can provide internet access to RMS devices.

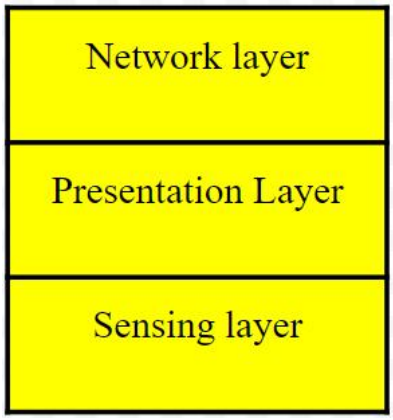

Figure 2:Generic RMS Device Architecture

TCP is a reliable end to end transport layer protocol. Transport Layer Security (TLS) and Secure Sockets Layer (SSL) are standard TCP based security protocols, which are based on hybrid combination of Symmetric and Asymmetric key encryption algorithm. Privacy and Data integrity of information with TLS and SSL comes with heavy layers computations and complex processing which is are not suitable for RMS devices which have limited computing power and other resources. These protocols incurs heavy headers and complex handshaking mechanism which again overburdens the processing unit [3]. Message Queuing Telemetry Transport [4] (MQTT) and Constrained Application Protocol [5] (CoAP) are the candidate IoT protocol for data transfer across the internet, these protocol have extremely lightweight headers. Both protocols uses underlying TCP and UDP transport layer to transfer the data. These protocols are standard internet protocol, which are heavily deployed over IoT devices. When massive IoT devices are deployed, all the devices are configured to a trusted party’s server application. These devices will be communicating with this application server

throughout their lives. So our work is directed towards developing a secured lightweight internet protocol between RMS IoT devices and trusted party's server application. Hence our work has adopted TCP layer to develop a scheme will help OEMs to deploy massive RMS IoT devices securely into various field application. The paper is divided into following sections- Sections II discusses the related work. Section III presents proposed security scheme and evaluates its performance and security analysis in Section IV.

II. Related Work

The work [6] provides a remote monitoring architecture for healthcare using sensor network over MAC protocol. In work [7] discusses the implementation of hardware and software design of the wireless RMS based on GSM. Work [8] presents an embedded database which is responsible for logging data with from sensor network and then syncing with remote database server using GSM/GPRS network. Work [9] & [10] presents RMS application with different technologies and architectures. In all the above work discussed, security of the information to be transferred is not addressed.

Work [11] has surveyed and briefed about the existing security protocol for IoT environment and also made a short comparison based upon the flaws and security aspects. Work [12] has compared MQTT with TCP socket by sending JSON data over both the protocol.

Author Jean Pierre Nzabahimana [13] have discussed and analyzed various security threats in IoT environment. Authors in [14] has identified securities threats and flaws in IoT devices from hardware perspective and also presented viable solution. Authors in work [15] has have discussed various glitches in Data Transport layer security (DTLS) and SSL Layer for IoT devices. Researchers in work [16] has discussed lightweight cryptographic algorithm and potential attacks on it. A Comparative analysis is also presented for their performance over windows and embedded hardware platform.

Motivation:

Considering all the above studies, our work is motivated towards developing an optimized and customized architecture for RMS IoT environment, which will comply constrained characteristics of IoT devices [17]. In this unique Identity Key (UIK) is hard-coded into IoT device at the time of manufacturing. Thus making end-to-end security strictly dependent on embedded cryptographic material, it reduces the complicated of handshaking procedures of key sharing.

III. Proposed Security Scheme

The Security Scheme "Object Secured Transmission Control Protocol" (OS-TCP) advocates using low sized header and lightweight encryption algorithm for secured data transfer. Every device is authenticated and then data is transferred to the trusted party's application server. Considering the constrained characteristics of IoT devices [17], the proposed scheme involves minimum handshaking to establish a secure session between application server and the Device. Cryptographic Material are embedded into the device's memory at the time provisioning and deployment phase. Further sections describes the implementations details of this scheme.

A) Object Security:

Object Security is more associated with the Application layer, it accentuates, securing the payload. 'Secured Object' in this work is referred as an independent entity where payload (actual message data) gets encrypted. Essentially it may consist of two components viz:

a) For data integrity, verification tags are calculated based on payload.
b) Metadata are introduces that describes security features of payload.

Payload field of any application protocol has this piece of encrypted information [18].

B) Key Management:

OS-TCP has an integrated key management where, only trusted party are involved in key exchange during the deployment and provisioning phase of the device. Key definitions are as follows:

1) Pre-Shared Key (PSK): PSK is 256 bit cipher key unique to each device.
2) Client ID (CID): Each device is uniquely identified by Application Server with CID. CID is represented by 10 byte number (ex. "999999001").
3) Pre-shared Authentication Number (PAN): PAN is 10 byte unique number (ex."1234567891") representing device used an authentication component. This Number is not sensitive to intruders.
4) Session Key (SK): SK is 256 bit cipher key, generated by the application server and it is valid for single TCP session.

PSK, CID and PAN are embedded into device's memory before deploying them over fields. These keys are shared with Application server's database where it is stored securely.

C) Lightweight Cryptographic Algorithm for secured TCP Socket:

Encrypt function has been formulated for OSTCP, which performs substitution–permutation (S-P) operation on plain data message and then a series of Exclusive - OR ($\oplus$) operations are computed on this S-P cipher text with as cipher key parameter. Cipher text received by Application Server is decrypted exactly in the reverse manner. Mathematical model for cryptographic algorithm is shown in Figures [8], [10].

D) Secured Scheme Protocol Packets:

To establish the secure session between the device and application server for data transfer, the scheme defines 5 types packet. PACKET-TYPE is one byte field that defines packet type. Timestamp of 12 bytes is introduced in every packet which includes date (DDMMMYY format) and time (HHMMSS format). Data Integrity and Transmission Errors are taken care by introducing Cyclic Redundancy Code (CRC) of 2 bytes.

Protocol Secured Scheme Packets are described as follows:

1) *Authentication Request Packet (PacketType-1):*

This Packet is sent whenever device wants to initiates data transaction. Each device should be authenticated by the for

data transfer phase figure 3 shows the Authentication Request Packet.

PACKET-TYPE(1)	DEVICEID	TIME-STAMP	CRC

Figure 3:Authentication Request Packet

2) *Authentication Status Packet (PacketType-2):*

After successful authentication by the server, Authentication Status Packet is sent to device shown in figure 4. Session Key is of 15 Bytes and valid for that particular TCP session. A-STATUS field is of 1 byte which indicates authentication status viz: 1 = Valid Client, 2 = Invalid Client, 3 = Transmission Error and 4 = Retransmit Request.

PACKET TYPE(2)	A STATUS	SESSION KEY	TIME-STAMP	CRC

Figure 4:Authentication Status Packet

3) Data Packet (PacketType-3):

After successful authentication, data is transferred over the TCP session with application server. PACKET-ID field is 5 bytes identifying each packet. Sensor Data depends upon the application and can go up to 1KB. Figure 5 shows Data packet.

PACKET TYPE(3)	PACKET ID	SENSOR DATA/MSG	TIME-STAMP	CRC

Figure 5:Data Packet

4) *Data Acknowledgement Packet (PacketType-4):*

Data Acknowledgement Packet is sent is response to Data packet by application server. DECRYPT-STATUS field is of 1 byte indicates the decryption status of data packet (PacketType-3). Viz: 1= data decrypted successfully, 2= Resend the packet, 3: Unknown Error. Figure 6 shows Data packet.

PACKET TYPE(3)	PACKET ID	DECRYPT STATUS	TIME-STAMP	CRC

Figure 6:Data Acknowledgement Packet

5) Disconnect Packet (PacketType-5): Disconnect Packet is sent by the device to server, after reception of this packet TCP session is terminated and the socket is disconnected.

PACKET-TYPE(5)	TIME-STAMP	CRC

Figure 7:Disconnect Packet

E) OS-TCP Protocol Phase:

OS-TCP secured protocol is divided into two phases which are follows.

1) Authentication Phase: Device sends an authenticate request packet to application server, to secure communication these steps are followed.

a) Device node produces cipher text payload (C) by computing C = Encrypt (Authenticate Request Packets, PSK).Where PSK is cipher key.
b) Cipher text payload (C) is appended (||) with PAN and then Authenticate packet is sent to application server.
c) Application server extracts PAN and uses as query parameter, then query is fired to server's database to find the corresponding PSK and CID. After successful retrieval of PSK and CID, server further decrypts the payload using PSK. Validation of timestamp in a tolerable range is done by the server by comparing with its own. CRC calculation are computed and validated against received one. After the successful qualification of the IoT device (client), AS sends Authentication Status packet with return code and Session key (SK) which is randomly generated by application server for that TCP session. This packet is encrypted the same way as discussed above (using PSK) and sent to the device node. Device Node extracts the SK after successful decryption.

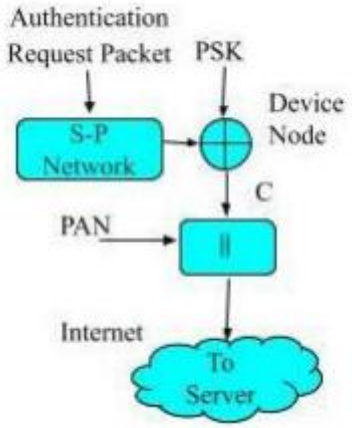

Figure 8:Authenticate Packet Encryption Mathematical Model

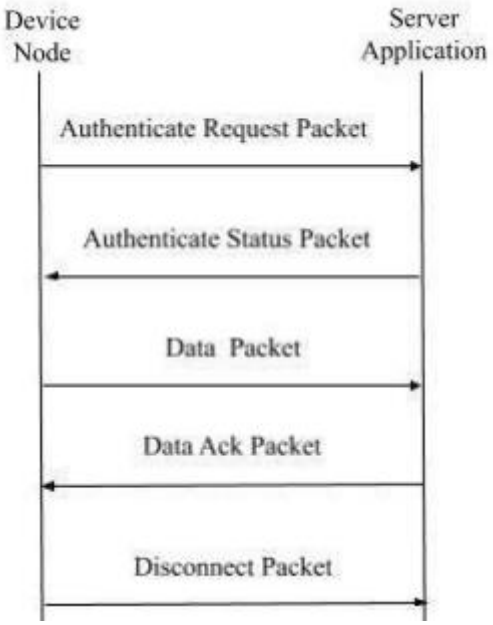

Figure 9:Proposed Protocol

2) *Data Transfer Phase:*

To secure this phase and strengthen the communication, 2 round with two different keys (PSK and SK) are computed to encrypt Data Packet and following steps are computed:

a) Device node produces cipher text payload (C1) for round 1 by computing C1= Encrypt (Data Packet, PSK). Where PSK is cipher key.

b) In second round cipher text (C2) is computed by C2 = Encrypt (C1, SK), where PSK is cipher key, and then Data packet is sent to application server.
c) Application Server then decrypts the Data packet using PSK and SK for that TCP session. After Successful decryption and validation (CRC and timestamp) of received Data Packet, a response is set to indicate the decryption status, Data Acknowledgment packet is sent to device with same Packet ID as received in Data packet.

F) Use Case:

RMS IoT device for Energy Meters An Energy meter device measures the amount of electric energy consumed by an electrical power device, buildings, hospital or a business. This Data produced by these meters need to be monitored by the energy billing centers. RMS device integrated with such meters can provide a perfect solution for end to end remote monitoring without human intervention, Figure 11 shows block diagram of RMS enabled Energy Meter. Data transmitted by such devices are more susceptible to hackers which can lead to fraudery in billing process, Thus OS-TCP scheme can be perfect lightweight solution to these meters [19].

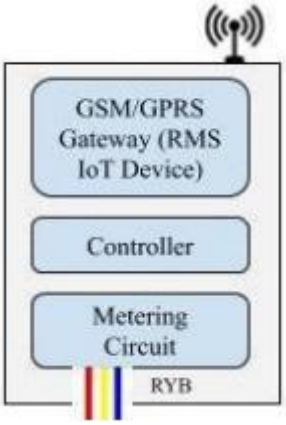

Figure 10:Energy Meter

IV. Performance Evaluation and Security Analysis

Performance evaluation depends upon the type of hardware architecture used in RMS device and availability of the network related resources. Real time performance of OS-MQTT is evaluated with the help of a generic RMS IoT device has been developed by R&D team at S.N. Systems Pvt. Ltd. This device used for accomplishing various network related research and rapid prototyping for IoT environment. Table 1 shows Hardware configuration of this RMS device. OS-TCP is implemented in Node.js environment. Node.js is scalable server side scripting in runtime JavaScript environment [21]. It has various packages for network applications which can be easily implemented. OS-TCP is implemented using NET package [22]. This script is deployed on public server for real time performance evaluation. Table 2 shows server configuration

Table 1: Hardware Configurations

GSM-GPRS Module	SimCom 868 [20]. (Core:-ARM7, ROM:-512K, RAM:- 448K and a small footprint of Linux based RTOS)
Power Supply	12V
Data Transmission	Mobile internet over 2G spectrum using GeneralPacket Radio Services(GPRS) Medium
Digital and Analog I/O	4 DI, 2 DO and 2 Analog I/P.

Table 2:Server System Configuration

Operating System	64 bit, Windows 2008 Server
Hardware	Intel(R)Xeon(CPU) X340 2.40 GHz
RAM	12 GB

RMS IoT device had shown following observations:
1) Successful registration to GSM and GPRS network, TCP socket initialization and connection to application server required, an average time of 20 seconds.
2) Cryptographic algorithm computation time is hardware architecture dependent, it will vary based hardware architecture. Table 3 shows the encryption timing for various data packet and size.
3) Round Trip Timing (RTT) for authentication process, i.e. device requesting and in response, getting authenticated status packet had been noted for 4 consecutive trial. Table 4 shows the respective timings for device.

Table 3:Encryption Time (in Millisecond)

Payload Size	50	100	200
Encryption Time (in milliSeconds)	1.040	1.6	2.15

4) Dummy data with different sizes has been sent to the application server, RTT is recorded for data sending frequency of 10 seconds for a time epoch of 5 min. Figure 12 shows the respective timings (Note: All the RTT timings are dependent on the availability of internet and network traffic).

Table 4:Authentication Request RTT (Millisecond)

Trail Count	1	2	3	4
RTT (mS)	418	464	488	502

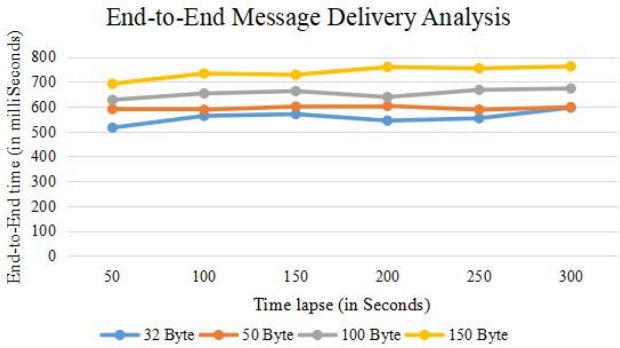

Figure 11:RTT Analysis at Device End

Security Analysis:

1. *Cloning and Man in Middle Attack Scenario with OSTCP.* Deployment and Provisioning phases involves trusted parties, exchanging PSK, PAN and CID with application server and RMS IoT device. These keys are stored securely with device's memory and server's database. After the successful authentication, server generates the session key (SK), both these keys ensures secured communication over the internet by protecting sensitive information. PAN are non-sensitive information, CID is encrypted and hence it obsoletes cloning attacks. Therefore any alteration of payload is not possible by the hackers, without keys.
2. *Replay Attack Scenario with OS-TCP.*

All the packets defined in OS-TCP have a timestamp embedded into it. Validates of timestamp is computed by the server application by comparing with its own system timings in a tolerable range. Hence terminated sessions cannot be initiated, because the packet will have stale timestamp and it will be immediately rejected.

CONCLUSION

OS-TCP is light weighted and minimum handshaking security scheme with customized and integrated key management. OS-TCP aims to put minimum burden over constrained IoT devices and offer reasonable security to information. The scheme has been implemented and analyze in real-time environment. The further Scope of our work includes designing and customizing the scheme for massive IoT deployments in various field applications.

V. ACKNOWLEDGEMENT

S. N. Systems Pvt. Ltd and Zhypility Technologies Pvt. Ltd has provided at most support to carry out this research. We appreciates our colleagues who provided expertise and insight which has greatly assisted our research work.

REFERENCES

[1] Hongping Fang; Kangling Fang; "The Design of Remote Embedded Monitoring System based on Internet" International Conference on Measuring Technology and Mechatronics Automation Year: 2010

[2] David Selvakumar; Kaushik Nanda; Hari Babu Pasupuleti; "Wireless Sensor Device Hardware Architecture- Design and Analysis for High Availability". 7th International Conference on New Technologies, Mobility and Security Year: 2015

[3] Carles Gomez, Andrés Arcia-Moret; Jon Crowcroft "TCP in the Internet of Things: From Ostracism to Prominence" IEEE Internet Computing Year: 2018

[4] Documentation MQTT protocol available online: http://mqtt.org/documentation

[5] Z. Shelby, B. Frank, and D. Sturek. Constrained Application Protocol (CoAP). Internet-Draft, Available at: https://tools.ietf.org/html/rfc7252. Year: June 2014.

[6] D. Benhaddou; M. Balakrishnan; X. Yuan "Remote Healthcare Monitoring System Architecture using Sensor Networks" IEEE Region 5 Conference, Year: 2008.

[7] Chen Pei Jiang, Jiang Xuehua "Design and Implementation of Remote Monitoring System Based on GSM" IEEE Pacific-Asia Workshop on Computational Intelligence and Industrial Application, Year: 2008.

[8] Jing Li ; Yong Xu, "Remote Monitoring Systems Based on Embedded Database" Third International Conference on Genetic and Evolutionary Computing, Year: 2009.

[9] E. Kanagaraj; L. M. Kamarudin; A. Zakaria; R. Gunasagaran; A.Y.M. Shakaff "Cloud-based remote environmental monitoring system with distributed WSN weather stations". IEEE SENSORS Year: 2015.

[10] Su Yang; Su Tong; Liu Liang, "Remote farm environment monitoring system based on embedded system and ZigBee technology" IEEE International Conference on Signal Processing, Communications and Computing (ICSPCC) Year: 2015.

[11] Snehal Deshmukh; S. S. Sonavane; "Security Protocols for Internet of Things: A Survey" 2017 International Conference on Nextgen Electronic Technologies: Silicon to Software (ICNETS2) Year: 2017.

[12] Neven Nikolov; "Research of the Communication Protocols between the IoT Embedded System and the Cloud Structure" IEEE XXVII International Scientific Conference Electronics - ET Year: 2018.

[13] Jean Pierre Nzabahimana, "Analysis of Security and Privacy Challenges in Internet of Things" The 9th IEEE International Conference on Dependable Systems, Services and Technologies, DESSERT Year: 2018.

[14] Subha Koley; Prasun Ghosal "Addressing Hardware Security Challenges in Internet of Things: Recent Trends and Possible Solutions", 2015 IEEE 12th Intl Conf on Ubiquitous Intelligence and Computing, Year: 2015

[15] Iulia Florea, "Challenges In Security in Internet of Things", 16th RoEduNet Conference: Networking in Education and Research (RoEduNet), Year: 2016

[16] Susha Surendran; Amira Nassef; Babak D. Beheshti "A Survey of Cryptographic Algorithms for IoT Devices" IEEE Long Island Systems, Applications and Technology Conference (LISAT) Year: 2018

[17] C. Bormann; "Terminology for Constrained-Node Networks" Internet Engineering Task Force (IETF) Available at: https://tools.ietf.org/html/rfc7228

[18] John Mattsson; "Object Security in Web of Things", Available online at: https://www.w3.org/2014/02/wot/papers/mattsson.pdf

[19] Qie Sun; Hailong Li; "A Comprehensive Review of Smart Energy Meters in Intelligent Energy Networks" IEEE Internet of Things Journal Year: 2016.

[20] SIMCom Wireless Solutions Ltd SIM868 Module‖ Available Online at: https://simcom.ee/modules/gsm-gprsgnss/sim868/

[21] Documentation for Node.js environment, available online at: https://nodejs.org/en/docs/

[22] Documentation for NET Node.js Package available online at:https://nodejs.org/api/net.html