2017

DENIAL THE DENIAL – SECURING 6LoWPAN AGAINST DOS

Mitigation strategies to minimize vulnerabilities in IoT 802.15.4 protocol stack

ABHISHEK MISHRA 2016BSY7508

ELL821 Research report – Under Prof. Ranjan Bose

INDIAN INSTITUTE OF TECHNOLOGY | Delhi

Table of Contents (Ctrl + Click to follow any link)

One page summary for the Project

Objective

To come up with a secure protocol suite enabling mitigation strategies against

1. Layer 2, 3, 4 Denial of service vulnerabilities
2. Identity spoofing attacks

How shall we do it?

Allow the master device to validate/invalidate a packet in just 16 bytes, no need of deep packet inspection. Proposed packet structure for this scheme:

16 bytes hash || AES encrypted packet (follows IPv6 packaging)

How are these hash bytes useful?

This 16 bytes hash is

1. Unique to each device and provides context to communication between the master and slave devices.
2. This provided context helps against identity spoofing, denial of service through flooding and packet replay attacks.
3. Changes every time master and slave devices talk (One-time pad's provable security).
4. Even on capturing packets, no critical information can be revealed.

This 16 bytes hash F_H is

F_H(RFD (slave) information, Sequence of packet, 16 bit right shifted unique identifier)

Sequence of packet and shifting of *unique identifier* are two variables which change every time.

Costs - Long term and short term

Short Term

1. This system is implementable in hardware as well as software and hence can be adopted easily.
2. Computation requirements are added for every packet validation (*Creation of hash for every transmission on RFDs (slave) and Creation of hash + Comparison of hash for every device on master device*).
3. These added computation requirements mean the master device needs to be a capable device whereas slave devices are relatively free of these.

Long Term

1. Once adopted this system will remove the bottleneck in IoT industry as a billion dollar business can't be built over an insecure model.
2. In critical systems where network layer vulnerability kills IoT implementation, this implementation is much cheaper as it allows sustainability in the long run.

State-of-the-Art advancements which exist

1. IEEE 802.15.4 doesn't propose any solution for denial of service, identity spoofing and other network layer vulnerabilities.
2. Key exchange is a challenge as no official protocols have been declared for it.
3. Identity verification can be bypassed easily in the current existing recommendations.

Limitations

1. So far only intra-PAN systems can be made secure with this proposed scheme.
2. In a global model of IoT, i.e where you need every device to be connected to every other device, it is not applicable. As it allows the slave devices to only talk to the master device.
3. Identity flexible inter-PAN models such as Vehicular IoT do not benefit from this scheme.

Literature Survey of Security of Internet of Things

Abhishek Mishra

I. INTRODUCTION

The Internet of Things is a jargon's word that refers to a proposed development of internet in which everyday objects have network connectivity. In simplistic manner one can state it as a grid in which every node can be potentially anything i.e a button from your shirt, a pen, a pacemaker, a watch etc. So you see that each of these nodes have the ability to communicate with other nodes and what we end up looking at is a world in which everything is connected. These connected nodes can be taught to function in a specific manner and thus our effort of connecting them could be put to some utility. This utility isn't specific but everything that you can imagine can be made possible with it but it is not free from the security and vulnerability issues that any network in modern world is. Even more so than any existing network we need to work extensively on its vulnerabilities as the network will comprise of low form factor, low power limitations and memory thus eliminating any intensive computational security of the modern world. This literature survey seeks to review the research of past 5 years in this field 2011-16 and find the school of thoughts which are common in them or contrasting.

The Internet of Things can be realized with either WSN (Wireless Sensor Networks) or M2M (Machine to machine) communications but either way we can't close our eyes to the various threats that still keep us from making IoT a reality[1]. The reason even after being introduced to the word Internet of Things in 1985 by Peter T. Lewis we still find people calling it a fancy term is that there is less consensus on how best to implement security in IoT at the device, network, and system levels . Firewalls and encryption techniques that aid and protect us against numerous threats to our privacy and security haven't been translated into embedded domain that well. So first we will address in section II what standards and systems have been created so far to address the issue of security vulnerability and other issues that comprise the IoT domain. In II we shall be looking forward at the thorough analysis of security vulnerabilities in these standards and models that exist. What we shall be limiting ourselves to are the IEEE models and standards that have been thoroughly documented and can be questioned and scrutinized and skip past the Open IoT architectures that do not endeavour to find a viable solution in the long run.

II. COMMUNICATION MODEL FOR INTERNET OF THINGS

In the discussion that lies in front of us we shall examine the protocol stack and layer model that IEEE research groups have worked and revised so far and suggest the areas in which security requirements are still lacking.

A. A layered model for IoT

Working groups such as Institute of Electrical and Electronics Engineers (IEEE) and the Internet Engineering Task Force (IETF) are designing new communications and security protocols that will play a fundamental role in enabling future IoT applications. These protocols are being designed in line with the low-energy sensing devices and low-rate data transfer needs and there have been few attempts[2] to give a complete stack for IoT but the standards community is taking its steps slowly and tackling the problems thoroughly. The protocol stack to be evaluated here is shown in fig. 1. From a bottom-up approach we shall mention key points of this protocol stack as follows:

- Low-energy communications at the physical (PHY) and Medium Access Control (MAC) layers are supported by IEEE 802.15.4[3]

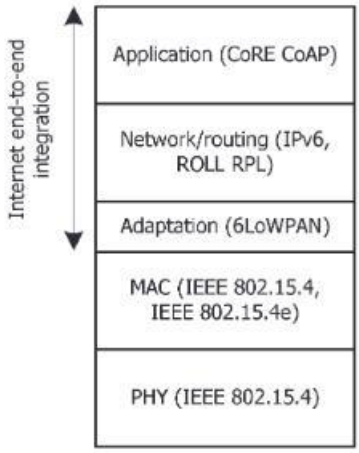


Fig. 1. Communication protocol stack for IoT

- Only 102 bytes are left for transmission of data at higher layers of stack whereas IPv6 needs a maximum transmission unit (MTU) of 1280 bytes and hence a research group was formed to devise a dedicated network layer protocol 6LoWPAN (IPv6 over low power wireless personal area network) [4]. 6LoWPAN also implements mechanisms for packet fragmentation and reassembly, among other functionalities.
- Routing over 6LoWPAN environments is supported by the Routing Protocol for Low-power and Lossy Networks (RPL)[5]. RPL is not only a routing protocol but it is a framework adaptable to IoT application domains as we understand that the nature of IoT is too generic to formulate a unadaptive routing system.
- The Constrained Application Protocol (CoAP)[6] supports communications at the application layer.

Given that all of these protocols are designed in lieu of IoT and its versatile application domains we shall examine and analyse IoT security requirements next.

B. A summary of IEEE 802.15.4[3], latest amendment

IEEE 802.15.4 specifications are as follows:

- operates at 2.4 GHz ISM band, 915 MHz and 868 MHz,
- uses BPSK, ASK and QPSK modulation techniques,
- standard data rate 250kbps with QPSK technique,
- output power: 20dbm (100 mW) at 2.4GHz, greater than 10dbm at 915 MHz, 30dbm at 868 MHz, MTU is 127 bytes per frame,
- consists of Fully Function Device (FFD) and Reduced Function Device (RFD)
  - FFD can talk to all types of devices and supports full protocol processing.
  - RFD can talk to only an FFD, lower power consumption and limited abilities.
- Devices are segregated into Personal Area Networks (PANs) fig. 2
  - Multiple PAN can operate on single channel.
  - Each PAN has a PAN identifier
  - Devices can communicate inter-PAN or intra-PAN
    1. PAN identifier is a 16 bit number and doesn't need any central authority to assign it.
    2. PAN identifier can be pre-assigned or assigned by the coordinator at start-up time.
- It is possible to scan multiple PAN identifiers on the same channel.
- Frames can be sent inter-PAN
- Broadcast PAN identifier is 0xFFF

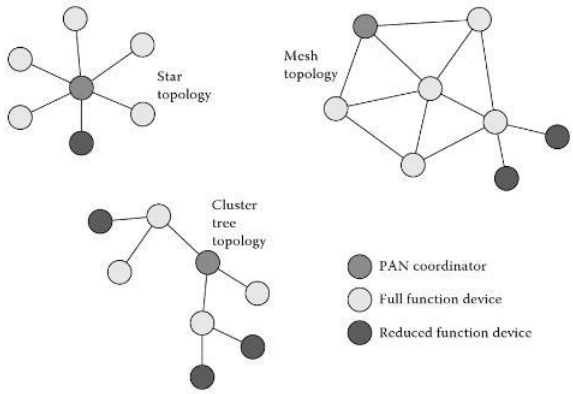


Fig. 2. Three topologies of LR-WPAN[7]

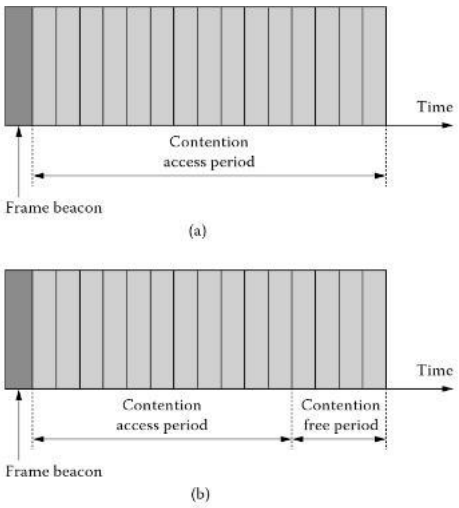


Fig. 3. Superframe structure (a) without GTSs (b) with GTSs

- Every device has two address - Short Address, a 16 bit PAN specific address, assigned by PAN coordinator and
- Long address, a 64 bit PAN specific address, assigned by a central authority, a global unique address.
- FFD is the coordinating device, processes request to join or leave the network and assigns short addresses to the devices.
- The network can be a beacon-enabled network optionally.
  - Coordinator sends a beacon frame to synchronise and delineate super-frames.
  - Access to the channel is time-slotted.
  - Super-frames can contain guaranteed time slots (GTS) each of which can be assigned to a specific device, preventing media access contention as shown in fig. 3
  - Beacon enabled networks consume less power as they have sleep mode.
  - In a beacon enabled superframe, frames must be sent in one of the slots, there are 16 total slots and one of them contain beacon superframe, see fig 4.
  - Slots in the contention-free period are each reserved for individual devices.
- Beaconless networks
  - No beacon frames transmitted by the coordinator.
  - Receivers must be listening all the time.
  - Full-time contention-access

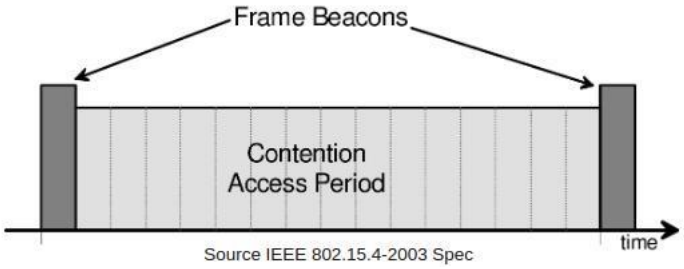


Fig. 4. Beacon-Enabled Network Superframe

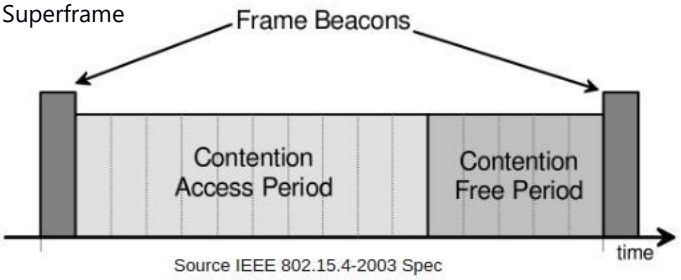


Fig. 5. Beacon-Enabled Network Superframe with Guaranteed Time Slots (GTS)

- Un-slotted and uses more battery, but easier to configure.
- Meshing is the ability to route messages through multiple hops on the network between source and destination.
- While 802.15.4 is designed with meshing in mind, it is not part of the 802.15.4 standard, and left to the network layer.

- Frame Types
  - Beacon Frame, sent by Coordinator to set up the Superframe structure.
  - Data Frame, transfers application data.
  - Acknowledgement Frame, provide confirmation of reception.
  - MAC Command Frame, Associate, Disassociate, Beacon request, GTS request
  - Data Frame Format , see fig. 7

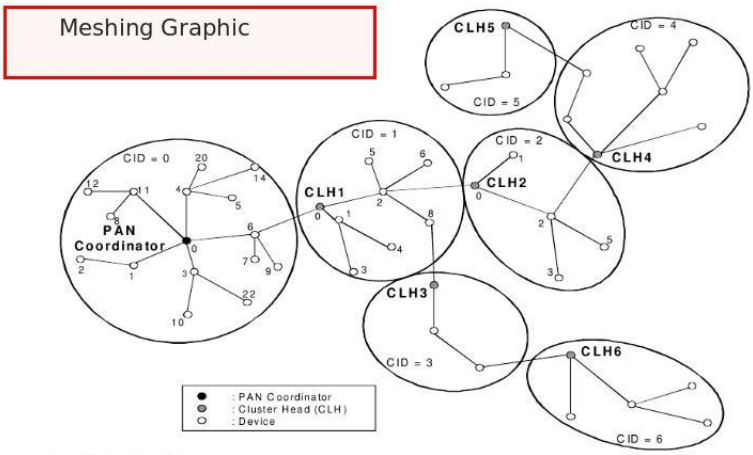


Fig. 6. Example Meshing Graphic

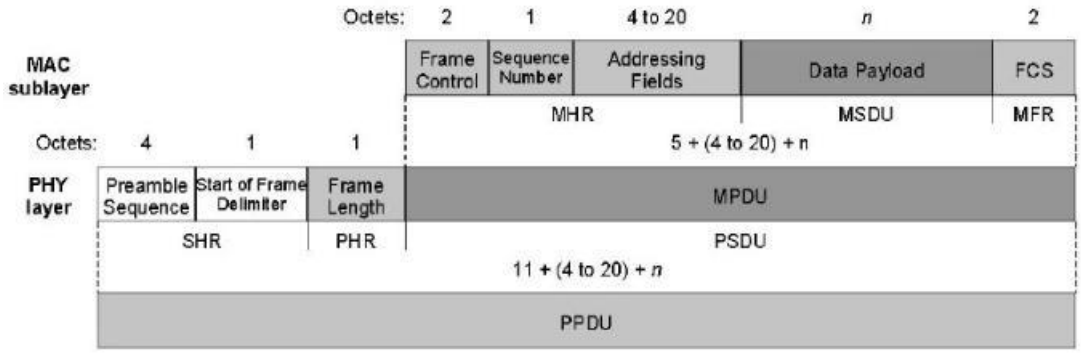


Fig. 7. Data frame format

- Collisions during data communications are managed in the Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) access method if the time-slotting mechanism for accessing FFD isn't applied.

C. Time-Synchronized Channel-Hopping MAC Layer Communications [8]

IEEE 802.15.4 standard may be unpredictable in terms of reliability, particularly in multi-hop usage scenarios, thus not being well suited to applications with restricted time constraints. As previously discussed, this is the case of applications in industrial environments currently supported by closed specifications such as WirelessHART and ISA 100.11a. With the goal of approaching this limitation, the recent IEEE 802.15.4e [3] addendum to the standard supports multi-hop communications using a technique originally proposed in the form of the Time Synchronized Mesh Protocol (TMSP) [9]

IEEE 802.15.4e channel hopping also requires synchronization between devices, which may be acknowledgment-based or frame-based. In the former, the receiver calculates the difference between the expected time of arrival of the frame and its actual arrival, and provides this information to the sender in the corresponding acknowledgment, thus enabling the sender to synchronize its clock to the clock of the receiver. In the latter, the receiver adjusts its own clock by the same difference, thus synchronizing to the clock of the sender. IEEE 802.15.4e also introduces a few modifications to the security services provided at the MAC layer, as we discuss later.

D. Security in IEEE 802.15.4 at MAC sub-layer

Current sensing platforms employing the cc2420 single-chip [10] RF transceiver from Texas Instruments, as the TelosB [11] mote from Crossbow, support IEEE 802.15.4 security and symmetric cryptography at the hardware using the Advanced Encryption Standard (AES) [12].

Security Modes: The IEEE 802.15.4 standard support various security modes at the MAC layer, which are described in fig. 8

Summarised AES implementation in IEEE 802.15.4e:

- Application decides the choices on the security level. (A bool value)
- Access Control Lists are used to enforce these security levels (max up to 255 entries).
- If security is enforced then the MAC layer looks up the ACL table for the cryptographic material for the destination.
- On packet reception, based on the flags the MAC layer decides how to process the packet, see fig. 9
- Description of security modes:
  - **NULL:** No security

Security mode	Security provided
No Security	Data is not encrypted Data authenticity is not validated
AES-CBC-MAC-32	Data is not encrypted Data authenticity using a 32-bit MIC
AES-CBC-MAC-64	Data is not encrypted Data authenticity using a 64-bit MIC
AES-CBC-MAC-128	Data is not encrypted Data authenticity using a 128-bit MIC
AES-CTR	Data is encrypted Data authenticity is not validated
AES-CCM-32	Data is encrypted Data authenticity using a 32-bit MIC
AES-CCM-64	Data is encrypted Data authenticity using a 64-bit MIC
AES-CCM-128	Data is encrypted Data authenticity using a 128-bit MIC

Fig. 8. AES Security Modes

Fig. 9. Format of an ACL entry in IEEE 802.15.4.

- **AES-CTR**: Encryption only, Counter Mode (Confidentiality alone).
  Break plain text into 16-byte blocks ($p_1, \ldots, p_n$). Compute cipher text $c_i = p_i$ xor $E_k(x_i)$
  CTR or Nonce x_i is necessary for the receiver to decrypt.

- **AES-CBC-MAC:** Message Authentication Codes (MAC) only (options of 32bit, 64bit and 128bit MAC's using Cypher Block Chaining (CBC).
- **AES-CCM:** Authenticated Encryption Mode (options of 32bit, 64bit and 128bit MAC using CTR and CBC MAC's).
- Replay protection can be turned on or off for any of the above.

E. Security for IoT Network Layer Communications

The IETF IPv6 over Low-power Wireless Personal Area Networks (6LoWPAN) working group was formed in 2007 to produce a specification enabling the transportation of IPv6 packets over low-energy IEEE 802.15.4 and similar wireless communication environments.

The characteristics of 6LoWPAN are as follows:

- Small packet size
- 16-bit short or IEEE 64-bit extended media access control addresses
- Low bandwidth. (250/40/20kbps).
- Topologies include star and mesh.
- Low power, typically battery operated
- Relatively low cost.
- Networks are ad-hoc and devices have limited accessibility and user interfaces
- Inherently unreliable due to nature of devices in the wireless medium

Bit Pattern	Short Code	Description
00 xxxxxx	NALP	Not A LoWPAN Packet
01 000001	IPv6	uncompressed IPv6 addresses
01 000010	LOWPAN_HC1	HC1 Compressed IPv6 header
01 010000	LOWPAN_BC0	BC0 Broadcast header
01 111111	ESC	Additional Dispatch octet follows
10 xxxxxx	MESH	Mesh routing header
11 000xxx	FRAG1	Fragmentation header (first)
11 100xxx	FRAGN	Fragmentation header (subsequent)

Fig. 10. Dispatch Codes in 6LoWPAN

6LowPAN Motivation: The pervasive nature of IP networks allows use of existing infrastructure and hence sticking to IP instead of any other technology has been preferred. IP-based technologies already exist, are well-known, and proven to be working though it has a few competitors like Recursive Internetwork Architecture (RINA)[13], IP still makes its mark and tools for diagnostics, management, and commissioning of IP networks already exist. IP-based devices can be connected readily to other IP-based networks, without the need for intermediate entities like translation gateways or proxies.

Challenges for 6LoWPAN:

Header Calculation

- IPv6 header is 40 octets, UDP header is 8 octets
- 802.15.4 MAC header can be up to 25 octets (null security)
- Or 25+21=46 octets (AES-CCM-128)
- With the 802.15.4 frame size of 127 octets, we have 127-25-40-8 = 54 octets (null security).
- For 127 – 46 – 40 - 8 = 33 octets (AES-CCM-128) of space left for application data. This much size for payload is a bottleneck in 6LoWPAN's implementation.

IPv6 MTU Requirements

- IPv6 requires links that support an MTU of 1280 octets.
- Link-layer fragmentation/ reassembly is needed.

6LowPAN Dispatch Codes: All 6LoWPAN encapsulated datagrams are prefixed by an encapsulation header stack and each header in the stack starts with a header type field followed by zero or more header fields. See fig. 10.

6LowPAN Frame Formats: Fig. 11 gives us worst case scenario for uncompressed IPv6/UDP. Dispatch code (01000001) indicates no compression Up to 54 / 33 octets left for payload with a maximum size and a MAC header with null / AES-CCM-128 security. The relationship of header information to application payload is obviously really bad in this scenario.

Fig. 12 gives us the best case scenario for compressed IPv6/UDP. Dispatch code (01000010) indicates HC1 compression. HC1 compression may indicate HC2 compression follows. This shows the maximum

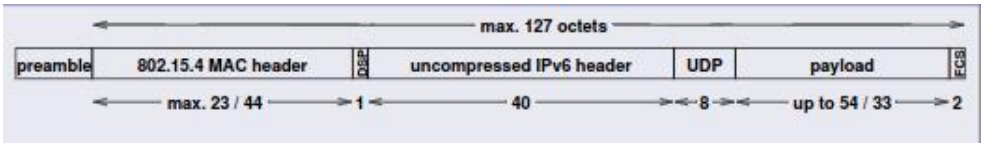


Fig. 11. Uncompressed IPv6/UDP (worst case scenario)

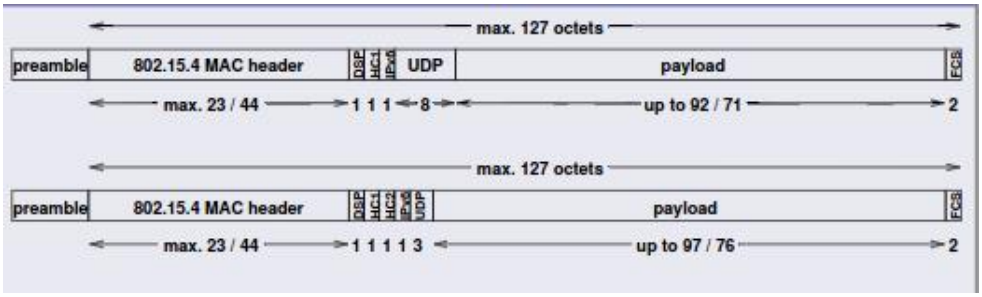


Fig. 12. Compressed Link-local IPv6/UDP (best case scenario)

compression achievable for link-local addresses (does not work for global addresses). Any non-compressible header fields are carried after the HC1 or HC1/HC2 tags (partial compression).

Security in 6LoWPAN: No security mechanisms are currently defined in the context of the 6LoWPAN adaptation layer, but the relevant documents include discussions on the security vulnerabilities, requirements and approaches to consider for the usage of network layer security, as we proceed to discuss.

The discussion regarding security on RFC 4944[14] is related to the possibility of forging or accidentally duplicating EUI-64 interface addresses, which may consequently compromise the global uniqueness of 6LoWPAN interface identifiers. This document also discusses that Neighbour Discovery and mesh routing mechanisms on IEEE 802.15.4 environments may be susceptible to security threats, and that AES security at the link-layer may provide a basis for the development of mechanisms protecting against such threats, particularly for very constrained devices. Other interesting discussion is on the possibility of employing more powerful 6LoWPAN devices in order to support heavy security related operations, also because such devices may support existing Internet security protocols, as such representing strategic places for the enforcement of security control mechanisms.

The discussion concerning security on RFC 6282[15] focuses on the security issues posed by the usage of a mechanism inherited from RFC 4944, which enables the compression of a particular range of 16 UDP port numbers down to 4 bits. This document discusses that the overload of ports in this range, if employed with applications not honouring the reserved set for port compression, may increase the risk of an application getting the wrong type of payload or of an application misinterpreting the content of a message. As a result, RFC 6282 recommends that the usage of such ports be associated with a security mechanism employing MIC codes.

III. CONCLUSION

In this literature review, a thorough analysis of IoT protocol stack (excluding Routing Protocol Layer (RPL), CoAP and Application Layer) has been done and there seems to be a weak defence in network layer against Denial of Service (DoS) attacks and fragmentation attacks as the design of 6LoWPAN has only focused on making IPv6 compatible with IEEE 802.15.4 and not really a robust solution has been proposed. On Logical Link Control and MAC sub-layer there is presence of AES and varying level of security is available by usage of different modes so the security at these sub-layers appears taken care of. The security at Physical layer has been left open and hence it is open to physical layer vulnerabilities in general.

REFERENCES

[1] Wind River. “Security in the Internet of Things”. In: Wind River Systems, Tech. Rep (2015).

[2] Maria Rita Palattella et al. “Standardized protocol stack for the internet of (important) things”. In: IEEE Communications Surveys & Tutorials 15.3 (2013), pp. 1389–1406.

[3] IEEE Standards Association et al. “IEEE Standard for Local and metropolitan area networks Part 15.4: Low-Rate Wireless Personal Area Networks (LR-WPANs), Amendment 4: Alternative Physical Layer Extension to Support Medical Body Area Network (MBAN) Services Operating in the 2360 to 2400 MHz Band”. In: (2013).

[4] Nandakishore Kushalnagar, Gabriel Montenegro, and Christian Schumacher. IPv6 over low-power wireless personal area networks (6LoWPANs): overview, assumptions, problem statement, and goals. Tech. rep. 2007.

[5] Tim Winter. “RPL: IPv6 routing protocol for low-power and lossy networks”. In: (2012).

[6] Carsten Bormann, Angelo P Castellani, and Zach Shelby. “CoAP: An application protocol for billions of tiny internet nodes”. In: IEEE Internet Computing 16.2 (2012), p. 62.

[7] Nandini Mukherjee, Sarmistha Neogy, and Sarbani Roy. Building wireless sensor networks. 1st Ed.

[8] Jorge Granjal, Edmundo Monteiro, and Jorge Sa´ Silva. “Security for the internet of things: a survey of existing protocols and open research issues”. In: IEEE Communications Surveys & Tutorials 17.3 (2015), pp. 1294–1312.

[9] K Pister and Lance Doherty. “TSMP: Time synchronized mesh protocol”. In: IASTED Distributed Sensor Networks (2008), pp. 391–398.

[10] 2016. URL: http://www.ti.com/lit/gpn/cc2420.

[11] 2016. URL: http://www.memsic.com/userfiles/files/Datasheets/WSN/telosb datasheet.pdf.

[12] Frederic P Miller, Agnes F Vandome, and John McBrewster. “Advanced Encryption Standard”. In: (2009).

[13] Gowtham Boddapati et al. “Assessing the security of a clean-slate internet architecture”. In: 2012 20th IEEE International Conference on Network Protocols (ICNP). IEEE. 2012, pp. 1–6.

[14] G Montenegro et al. “RFC 4944”. In: Transmission of IPv6 packets over IEEE 802.4 (2007).

[15] Jonathan Hui and Pascal Thubert. RFC 6282 compression format for ipv6 datagrams over IEEE 802.15. 4-based networks. Sep-2011. 2011.

6LoWPAN vulnerabilities and counter-measures

Note: Here I have listed researched vulnerabilities and attacks that exist in 6LoWPAN (IoT Network Layer). The purpose of listing them is to create a testing framework for our proposed solutions and it also acts as a guiding tool in creation of experimental setup.

6LoWPAN Vulnerabilities – at a glance (Expansion possible) [1]

1. Information Disclosure
2. User and Administrative CLI (Command Line Interface)
3. Denial of service
4. Unencrypted services (if any)
5. Poor encryptions (if any)
6. Test / Development environment
7. Buffer overflow
8. UPnP (Universal plug-n-play)
9. Vulnerable UDP services
10. Device firmware OTA update block
11. Replay attack
12. Lack of payload verification
13. Lack of message integrity check
14. Credential management vulnerabilities
    - Username authentication
    - Weak passwords
    - Account lockout
    - Known default credentials
    - Insecure password recovery mechanism

6LoWPAN Attacks – at a glance (Expansion possible) [2]

1. IP Spoofing
2. RIP(Routing Information Protocol) attack
3. ICMP (Internet Control Message Protocol) attack
4. ARP spoofing
5. PING Flood (ICMP Flood)
6. Ping of Death Attack
7. Teardrop Attack
8. Packet Sniffing

The biggest vulnerability – Distributed Denial of Service (DoS) [3]

Possible DDoS Traffic types

Assuming the application layer protocol is referred to as **ALP (e.g. HTTP)** we shall be mentioning the possible DDoS attacks, categorised layer-wise, as follows:

Type	Description
ALP Flooding	ALP Header flood/request ALP POST flood/request ALP GET flood/request
SYN Flooding (TCP/SYN)	SYN Flood works by establishing half-open connections to a node. When the target receives a SYN packet to an open port, the target will respond with a SYN-ACK and try to establish a connection. However, during a SYN flood, the three-way handshake never completes because the client never responds to the server's SYN-ACK. As a result, these "connections" remain in the half-open state until they time out.
UDP Flooding	UDP floods are used frequently for larger bandwidth DDoS attacks because they are connectionless and it is easy to generate protocol 17 (UDP) messages from many different scripting and compiled languages.
ICMP Flooding	Internet Control Message Protocol (ICMP) is primarily used for error messaging and typically does not exchange data between systems. ICMP packets may accompany TCP packets when connecting to a server. An ICMP flood is a layer 3 infrastructure DDoS attack method that uses ICMP messages to overload the targeted network's bandwidth.
MAC Flooding	A rare attack, in which the attacker sends multiple dummy Ethernet frames, each with a different MAC address, Network switches treat MAC addresses separately, and hence reserve some resources for each request. When all the memory in a switch is used up, it either shuts down or becomes unresponsive. In a few types of routers, a MAC flood attack may cause these to drop their entire routing table, thus disrupting the whole network under its routing domain.

Solutions to above-mentioned DDoS attacks

ALP Flooding [4] Application-layer DDoS attacks are a bit more complicated. Layer 7 DDoS attacks are some of the most difficult attacks to mitigate against because they mimic human behavior as they interact with the user interface. A sophisticated Layer 7 DDoS attack may target specific areas of a website, making it even more difficult to separate from normal traffic. For example, some types of Layer 7 DDoS attacks will target website elements, like your logo or a button, and repeatedly download resources hoping to exhaust the server. Still another example is when an attacker targets a download on a website and proceeds to go through the process the author just described above.

How To Fight A Layer 7 DDoS Attack?

- Block spoofed TCP attacks before they enter your network.
- Don't let dark address packets pass your perimeter.
- Block unused protocols and ports.
- Limit the number of access per second per source IP.
- Limit numbers of concurrent connections per source IP.
- Filter foreign TCP packets.
- Do not forward packets with header anomalies.
- Monitor self-similarity in traffic.
- Keep unwanted guests away.
- Use specialized DDoS mitigation equipment.

SYN Flooding [5] TCP connections take place in three stages (commonly known as the three-way handshake). A SYN flood takes advantage of this process by sending an initial SYN request to a server, but never completing the final stage of the connection process, essentially leaving the server "hanging" with a half-open connection. If all of a server's available connections are stuck in this half-open state, there's no more room for legitimate requests to be processed, and the server can appear down.
Such kinds of attacks can be mitigated by implementing a memory-allot-on-ack scheme. In memory-allot-on-ack scheme we initiate a connection and simply store it in a software and the memory allotment or wait for response process is only initiated after the connection is validated. Once the connection is validated the waiting for response process is valid only for a small interval of time and idle connections are terminated.

UDP Flooding/ICMP Flooding [5] This DDoS attack leverages the User Datagram Protocol (UDP), a session-less networking protocol. This type of attack floods random ports on a remote host with numerous UDP packets, causing the host to repeatedly check for the application listening at that port, and (when no application is found) reply with an ICMP Destination Unreachable packet. This process saps host resources, and can ultimately lead to inaccessibility. Similar attack is ICMP flood where random control information keeps on being sent to a host node in order to exhaust it's resources.
This is a very tough attack to mitigate against and special measures need to be taken in order to deal with it. It can only be dealt by somehow identifying the flooding IP and instructing the packet reader to reject the packet as soon as its source IP is identified.

MAC Flooding In order to deal against MAC Flooding there has to be applied a combination of ACL (Access Control Lists) and capabilities list. The idea is to come up with a behavioural expectation of different classes of IPs in network, flagging them upon breach of such and finally barring your neighbour network from forwarding messages from the malicious node to the host.

Talking about things in context of IoT

What is the context of IoT? To be honest, nobody can answer that. Millions of devices spread everywhere, to be expected anywhere and embedded in everything. If you aim to come up with a generic definition which claims to solve problems for all problems for all kinds of IoT application then it is only waiting to be

proven wrong somewhere in the future. I do not aim to say that there is no solution possible but that saying that you have a solution adaptable for everything is claim of finding a "panacea".
The solution that we should aim to provide is something that can be presented with a POC (Proof of Concept), deals with an application of IoT like vehicular systems, smart homes, wearable IoT or industrial IoT and can be proven to be scalable with having affordable implementation on all tiers.

Security in 6LoWPAN

So far through the discussions that have been done to secure 6LoWPAN we have focussed on Smart Home model and are aiming towards securing it from the attacks as mentioned in section 1 in this write-up. Our idea is trying to take advantage of the fact that one may not be able to completely ward-off all attempts of communication from any malicious party but through reducing the time that it takes for us to reject the communicated packet we don't bother our throughput performance by much and can react in time by raising an alarm situation by analysing attack-signatures.

How would you reduce time taken to reject a packet in case it is an unauthenticated source?

How do you know if a source is authenticated in Smart Home scenario?

- An RFD (Reduced function device) is authenticated as it is assigned with a 16 bit unique identifier that has been allotted to it by the FFD (Fully Functional Device) at the time of it being admitted into the PAN.
- The FFD keeps a list of authenticated IPs and only allows them to make any request. If it realises that an RFD which doesn't have the EUID is trying to contact then it shall drop the connection.
- All connections are recommended to be TCP connections.

How can you be sure that the unique identifier can't be spoofed via packet replay attacks?

- It can't, as the information that is passed on to the FFD is actually valid for one time only.

How shall the FFD know if it is changing every time? Isn't the purpose of FFD to be able to identify correct packet sender?

- The proposed packet structure is this.

CRC32 || Cryptohash (16 bytes) || AES Encrypted packet (contains headers and payload data)

- CRC32 [6] – For a 127 bytes packet (current packet size proposed in 6LoWPAN) it has a success rate of 99.999% in determining whether the packet is corrupted or not.
- Cryptohash – A 16 bytes cryptographic hash value that has been generated from input. Input is your 16 bit unique identifier, sequence of the packet, RFD information (Manufacturer, 64 bit unique identifier issued by a trusted source).
- The serial no. of the packet keeps on changing and hence the final cryptographic hash keeps on changing as well.
- Both the RFD and the FFD are aware of this as FFD has allotted the 16 bit EUID and it knows the sequence of the packets. The RFD information has been shared with the FFD in the time when it was first admitted into the system.
- AES encrypted packet – contains headers and payload.

How would you authenticate devices? How would you ensure you don't admit a malicious device to the system?

- The admission of a device in a system is done in this process:
    1. The FFD doesn't receive runtime installation requests instead an interrupt has to be raised by the users. The author recommends use of a remote RFD to manually select **install mode.**
    2. The remote RFD is a device with a static IP address that the FFD has allocated it. It is a trusted device and communicates using digital signatures which the FFD is capable of verifying.
    3. Upon receiving the interrupt to enter **install mode** the FFD finishes ongoing actions and broadcasts the message to all RFDs telling them to go to **sleep mode.** The FFD is now ready to drop all packets that don't follow the packet format as mentioned below.
    4. The RFD to be authenticated/installed has to send a packet of this format to FFD

        CRC16 || 16 bytes set to 0's || AES encrypted packet (contains all the RFD information)

    5. The FFD receives this message, decrypts the packet and communicates all this information to the remote RFD. The remote RFD displays this information to the user who verifies that this information is correct and approves it with his passphrase/pattern/key.
    6. The FFD shall now communicate a 16 bit unique identifier to the RFD which will be then used by the RFD in subsequent communications.
    7. The FFD shall exit **install mode** to enter **master mode.**

How would the subsequent communications go on?

- Each RFD is allotted a time slot in which they're supposed to communicate with FFD.
- The RFD shall create a hash of 16 bytes using this format:

$$F_H(\text{RFD information, Sequence of packet, 16 bit right shifted unique identifier})$$

- F_H – A hashing algorithm, possible candidates are **Spongent** [7], **Quark** [8]**, Photon** [9]**, Keccac** [10]**, Present family** [11] of hashing algorithms.
- The RFD sends the packet and the FFD receives it. Once it has received a packet, it calculates its own hash of 16 bytes and compares it with the hash received. If they match, it processes the packet for whatever purpose it was sent and if it doesn't match then it erases the packet.
- Once it has accepted the packet it was waiting for in that time slot it refuses any packet that comes afterwards in the same time slot.
- In the next time slots, devices shall perform same actions for their corresponding communications.

How are the cryptographic hashes created and their statistics?

- Lightweight hashes that qualify for usage in 6LoWPAN devices pertaining to this design are
    1. Spongent family
    2. Photon family
    3. Quark family
    4. Keccak
    5. Present
- A comparison of all of them is shown below:

Hash function	Parameters[a]			Security		Area[b]	Lat.	Thr.	Power [µW]	
	n	c	r	Pre	Col	[GE]	[cycles]	[kbps]	Mean	Peak
u-Quark	136	128	8	128	64	1379	544	1.47	2.44	2.96
u-Quark×8	136	128	8	128	64	2392	68	11.76	4.07	4.84
d-Quark	176	160	16	160	80	1702	704	2.27	3.10	3.95
d-Quark×8	176	160	16	160	80	2819	88	18.18	4.76	5.80
s-Quark	256	224	32	224	112	2296	1024	3.13	4.35	5.53
s-Quark×16	256	224	32	224	112	4640	64	50.00	8.39	9.79
Implementations of present-based hashes from (0.18 µm)										
dm-present-80	64	64	80	64	32	1600	547	14.63	1.83	-
dm-present-80	64	64	80	64	32	2213	33	242.42	6.28	-
dm-present-128	64	64	128	64	32	1886	559	22.90	2.94	-
dm-present-128	64	64	128	64	32	2530	33	387.88	7.49	-
h-present-128	128	128	64	128	64	2330	559	11.45	6.44	-
h-present-128	128	128	64	128	64	4256	32	200.00	8.09	-
Implementations[c] of Keccak from [15, §9.4] (0.13 µm)										
Keccak[72,128]	200	128	72	128	64	1300	3870	1.86	-	-
Keccak[40,160]	200	160	40	160	80	1300	3870	1.03	-	-
Implementations of Keccak from (0.13 µm)										
Keccak[72,128]	200	128	72	128	64	2520	900	8.00	5.60	-
Keccak[72,128]	200	128	72	128	64	4900	900	400.00	27.60	-
Keccak[40,160]	200	160	40	160	80	2520	900	4.44	5.60	-
Keccak[40,160]	200	160	40	160	80	4900	900	222.22	27.60	-

Table 1 Comparison of Quark, Present and Keccak

Implementations of photon from (0.18 µm)										
photon-128/16/16	128	128	16	112	64	1122	996	1.61	2.29	-
photon-128/16/16	128	128	16	112	64	1708	156	10.26	3.45	-
photon-160/36/36	160	160	36	124	80	1396	1332	2.70	2.74	-
photon-160/36/36	160	160	36	124	80	2117	180	20.00	4.35	-
photon-224/32/32	224	224	32	112	64	1736	1716	1.86	4.01	-
photon-224/32/32	224	224	32	112	64	2786	204	15.69	6.50	-
Implementations of spongent from (0.13 µm)										
spongent-128	128	128	8	120	64	1060	2380	0.34	2.20	-
spongent-128	128	128	8	120	64	1687	70	11.43	3.58	-
spongent-160	176	160	16	144	80	1329	3960	0.40	2.85	-
spongent-160	176	160	16	144	80	2190	90	17.78	4.47	-
spongent-224	240	224	16	208	112	1728	7200	0.22	3.73	-
spongent-224	240	224	16	208	112	2903	120	13.33	5.97	-

Table 2 Additional comparison of Photon and Spongent

Reference of data for Tables [8]

a - For the non-sponge present-based functions, n, c, r are respectively the lengths of a digest, internal state, and message block.

b - For Quark implementations, one GE is the area of a 2-input drive-one NAND gate, i.e., in the target 0.18 µm technology, 9.3744 μm^2

c - These implementations use external memory to store the internal state.

We aim to compare the results with all of them one by one. In the beginning we shall prefer to start with the one with code that can be most easily available.

How would you create a testing framework for your proposed model?

- The testing framework will be first tested with a discrete event simulation. The discrete event simulation shall be created in *C Language*.
- No additional libraries or proprietary material is needed for its implementation on current model.
- Once the implementation difficulties are experienced and better dealt with in simulations, next aim is to implement the same on hardware and come up with recommendations for device specifications that can successfully run it.

Results of Simulation

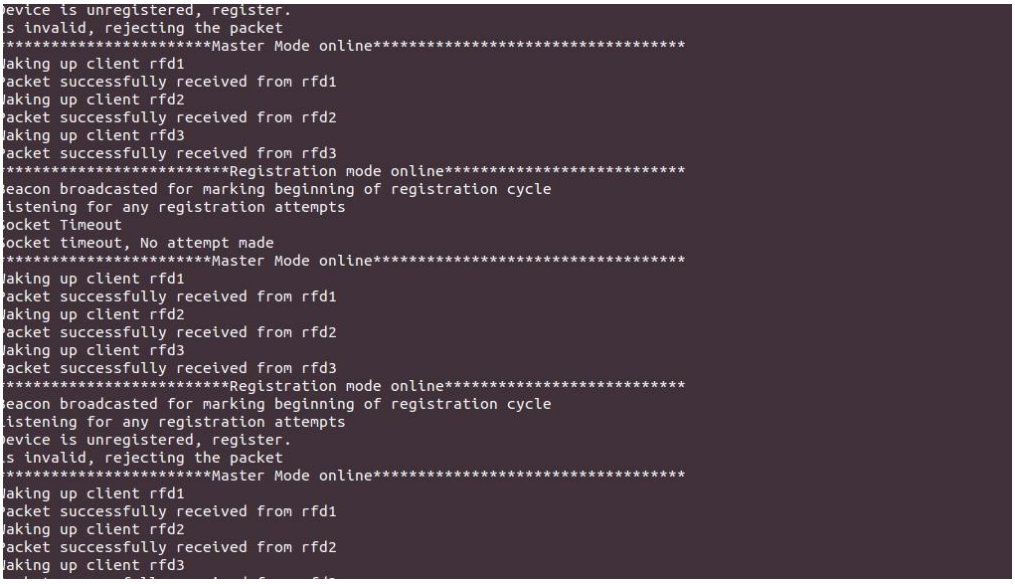
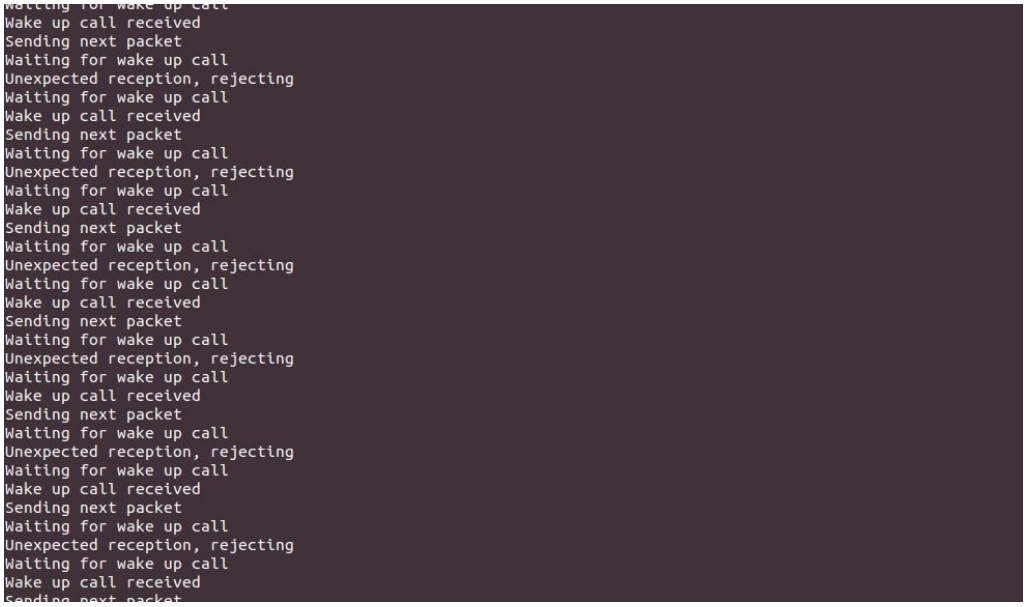


Figure 1 Master Device

Figure 2 Slave Device



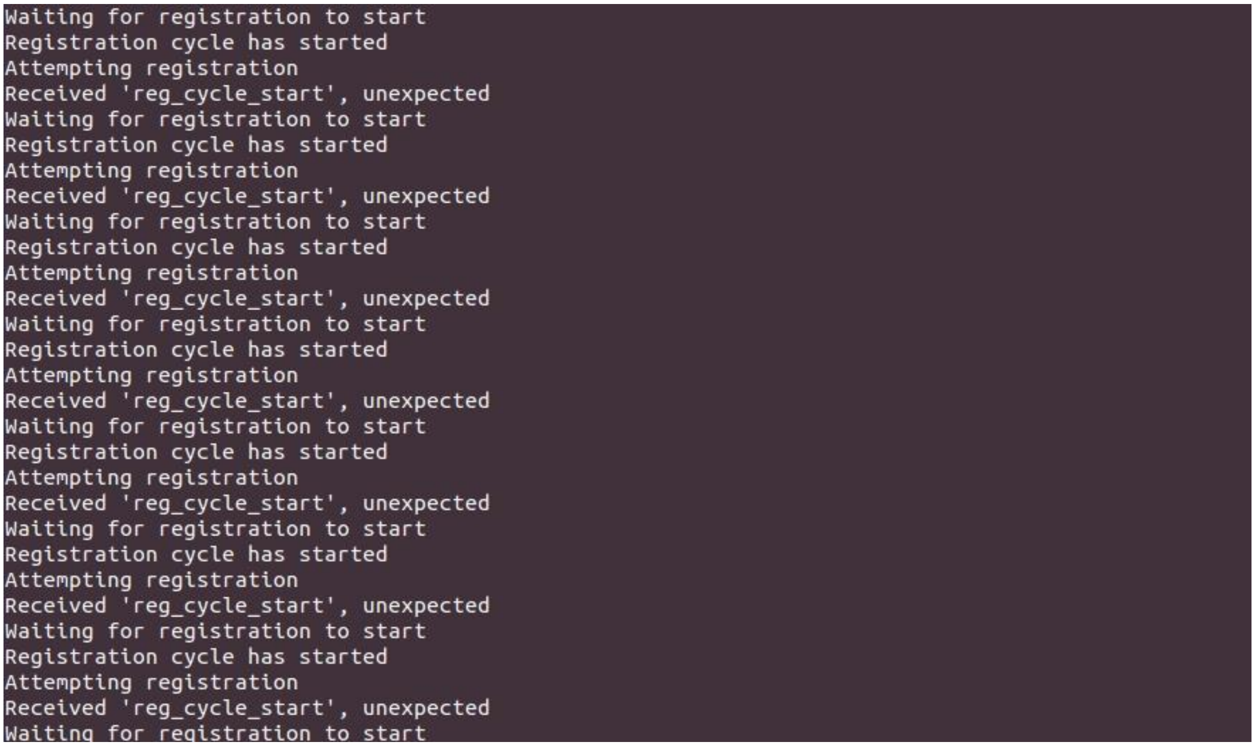
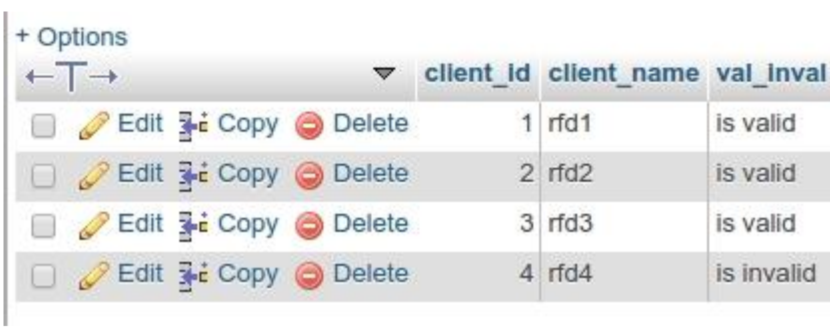


Figure 3 Malicious slave party

#	Name	Type	Collation	Attributes	Null	Default	Extra	Action
1	cli_id	int(2)			No	*None*	AUTO_INCREMENT	Change Drop Primary Unique Index Spatial Fulltext More
2	cli_name	varchar(8)	latin1_swedish_ci		No	*None*		Change Drop Primary Unique Index Spatial Fulltext More
3	sequence	int(2)			No	0		Change Drop Primary Unique Index Spatial Fulltext More
4	euid	int(16)			No	*None*		Change Drop Primary Unique Index Spatial Fulltext More
5	info	varchar(80)	latin1_swedish_ci		No	*None*		Change Drop Primary Unique Index Spatial Fulltext More
6	address	int(8)			No	*None*		Change Drop Primary Unique Index Spatial Fulltext More
7	idle	int(2)			No	0		Change Drop Primary Unique Index Spatial Fulltext More

Figure 4 Database register



+ Options

	client_id	client_name	val_inval
Edit Copy Delete	1	rfd1	is valid
Edit Copy Delete	2	rfd2	is valid
Edit Copy Delete	3	rfd3	is valid
Edit Copy Delete	4	rfd4	is invalid

Figure 5 Trusted Authority list

References

[1] "OWASP Internet of Things project," [Online]. Available: https://www.owasp.org/index.php/OWASP_Internet_of_Things_Project.

[2] D. K. G., "Network Security Attacks and Countermeasures," in *Network Security Attacks and Countermeasures*.

[3] "DDos Quick Guide," [Online]. Available: https://www.us-cert.gov/sites/default/files/publications/DDoS%20Quick%20Guide.pdf.

[4] "Layer 7 DDOS Attacks," [Online]. Available: http://ddosattackprotection.org/blog/layer-7-ddos-attack/.

[5] "DDos Attacks Incapsula," [Online]. Available: https://www.incapsula.com/ddos/ddos-attacks/.

[6] J. G. M. P. C. H. J. Stone, "Performance of Checksums and CRCs over real data," *IEEE/ACM Transactions on Networking,* vol. Volume 6, no. 5, p. 529 – 543, Oct 1998.

[7] A. Bogdanov, "Spongent: The design space of lightweight cryptographic hashing," *IEEE Transactions on Computers,* 2013.

[8] J.-P. Aumasson, "Quark: A lightweight hash," *International Workshop on Cryptographic Hardware and Embedded Systems.*

[9] J. T. P. a. A. P. Guo, "The PHOTON family of lightweight hash functions," in *Annual Cryptology Conference*.

[10] E. B. a. T. Y. Kavun, "A lightweight implementation of Keccak hash function for radio-frequency identification applications," in *International Workshop on Radio Frequency Identification: Security and Privacy Issues*.

[11] A. Y. Poschmann, "Lightweight cryptography: cryptographic engineering for a pervasive world," 2009.