Power-Aware Security Protocols for the Internet of Things

Tiago Miguel Correia Diogo

Instituto Superior Técnico, Universidade de Lisboa
Avenida Rovisco Pais 1, Lisboa,
tiago.diogo@tecnico.ulisboa.pt

Abstract. le pretty abstact...

Keywords: power-aware, security, protocols, internet of things

Table of Contents

1 Introduction

The Internet of Things (IoT) can be seen as web of interconnected devices that go from everyday wearable objects into fully deployed sensor networks.Despite the huge variety and characteristics of these devices, one thing that they all have in common in the constrained nature they're built upon. In order to enable the massive deployment to be expected in the near future [1] IoT devices must be accessible and affordable, capable of operating under lossy wireless networks while being battery powered. This poses a challenge to current Internet protocols since the assumptions regarding the devices capabilities and objectives do not hold true. To allow the IoT vision to come forward, several new protocols have been developed across the OSI layers, each addressing and tackling the challenges involved in trying to keep the quality and assurances of stronger, more expensive protocols, on constrained systems. Additionally, security is also a big topic of interest due to the fact that the interconnection of the devices around us can provide information about our choices and whereabouts, therefore reducing our privacy. This document will address these issues from a power-aware perspective, meaning the battery consumption will be of major importance.

2 Main Goals

Given the constrains and limitations of IoT devices described in the previous chapter and the recent protocols, the first objective of this work is to identify a working stack of protocols that takes into account these constraints and focuses on allowing a power-aware communication model.

After the analysis of the existing solutions, a baseline of power consumption will be established. Then, the focus will move towards adding confidentiality to the transmitted information by securing the channel. Once both the application level protocols and proper security solutions are defined, experiments will be performed so that the added power consumption cost of adding the security layers can be measured, profiled and documented therefore enabling the finding of the best parameters for a desired level of security.

The work will then proceed towards finding effective counter-measures against a specific group of attacks that targets the IoT devices by intensifying the use of its resources therefore draining the available power and placing the device offline. The ultimate goal is to propose an energy-efficient security mechanism that can resist these power-drain (a.k.a vampire) attacks.

[1] http://blogs.wsj.com/cio/2015/06/02/internet-of-things-market-to-reach-1-7-trillion-by-2020-idc/

3 Related Work

3.1 Protocol Analysis and Selection

There are many alternatives and some proposed standards when it comes to choosing a protocol stack for IoT communications. The decision must be based on the particularities of the devices to be used and the objective of the application itself, however a thoroughly analysis of the existing solutions is a proper way to unveil the strong and weak points of each protocol providing a good basis for an informed decision. A recent survey (January 2015) [1] of the IoT enabling technologies, protocols and applications will be the starting point for the analysis to follow. The presentation of the available protocols and solutions will follow a bottom-up approach, starting from the data link and physical layer all the way up until the application layer. In particular, the session layer will be left to the end since securing the channel is optional and will be addressed after the application level protocols are properly examined.

3.1.1 Data Link and Physical Layer

The first requirement for the physical layer of the IoT is the use of wireless radios. These should aim for simplicity, low-power and low-cost communications. While wireless communication are far spread and can be found from homes to airports, the type of radio commonly used, known as Wi-Fi, use a high amount of power causing concerns for battery life. In the next paragraphs, an overview of Wi-Fi(IEEE 802.11) is given with the objective of comparing it with the IEEE 802.15.4, a protocol that aims to address these issues.

IEEE 802.11

IEEE 802.11 is a set of standards for Wireless Local Area Networks (WLAN) communications. They are the basis for the so called Wi-Fi. IEEE 802.11 is concerned with Ethernet matching speed, long ranges, message forwarding and high data throughput. These concerns directly clash with the IoT objectives and account for the added power consumption of this protocol.

IEEE 802.15.4

IEEE 802.15.4 on the other hand was created for Low-Rate Wireless Private Area Networks (LR-WPAN) and its specifications on low power consumption, low data rate, low cost and high message throughput make it a strong candidate for IoT applications. The IEEE 802.15.4 standard supports two types of network nodes, the Full Function Device (FFD) that act as coordinator or normal nodes. And the Reduced Function Device (RFD) that are very simple, with very restricted resources and can only communicate with coordinators. The coordinators are responsible for controlling and maintaining the network. FFD are capable of storing a routing table in their memory and can implement a full Medium

Access Control (MAC). IEEE 802.15.4 supports star, peer-to-peer(mesh) and cluster-tree topologies. Regarding performance, it would be unfair to directly compare the two, since IEEE 802.11 transmission power and receiver sensitivity and much greater than 802.15.4. But if we limit both to a low power level IEEE 802.11 still outperforms IEEE 802.15.4 in terms of packet delivery ratio, throughput, latency, jitter and and average energy consumption. However this comes at the cost of a far lower transmission range[2]. We can conclude that for typical LR-WPAN network requirements, IEEE 802.15.4 is better designed to address the constrained environment issues, while IEEE 802.11 would still be a suitable option if transmission range is not a problem.

3.1.2 Network Layer

6LoWPAN

The IoT vision, as presented in the introduction, and its massive deployment can only be achieved through the use of IPv6. However, physical layers more suitable for communication over constrained networks pose some limitations to the use of the IPv6 messages. For example the limited packet size in IEEE 802.15.4 based networks. To tackle these issues, the Internet Engineering Task Force (IETF) 6LoWPAN working group developed a standard based on header compression to reduce the transmission overhead, fragmentation to meet the IPv6 Maximum Transmission Unit (MTU) requirements and forwarding to link-layer to support multi-hop delivery. [3] 6LoWPAN is able to remove a major share of IPv6 overheads, being able to compress its headers to two bytes, therefore allowing small IPv6 datagrams to be sent over IEEE 802.15.4 networks.

RPL

With the use of 6LoWPAN, upper layer routing protocols can now use the IPv6 addressing scheme. Given the possible frequent topology changes associated with the radio-link instability, successful solutions must take these requirements into account on their specification. RPL can support a wide variety of link-layers and is prepared for devices with very limited resources. It is able to build up network routes, distribute routing knowledge among nodes and adapt the topology in a very efficient way.

3.1.3 Application Layer

Hypertext Transfer Protocol (HTTP)

HTTP is an application level protocol that works in the request-response model and is the foundation of data communication on the World Wide Web (WWW) It is primarily designed to run over Transmission Control Protocol (TCP) which is a problem in lossy and constrained environments due to the delivery assurances

and congestion control algorithms it employs. Besides, HTTP is verbose, text-based, and not suited for compact message exchanges. Moreover, the header size required for a message exchange can leave too few payload space in constrained networks like the IEEE 802.15.4-based networks where the MTU size of the protocol is 127 bytes. These protocol specifications would not raise any issues in standard WWW communications, but when it comes to constrained environments it is clear that the protocol is not adequate to the necessities of IoT devices and networks.

Constrained Application Protocol (CoAP)

CoAP is a document transfer protocol based on REpresentational State Transfer (REST) on top of HTTP functionalities. CoAP objective is to enable tiny constrained devices to use RESTful interactions, where clients and servers expose and consume web services using Universal Resource Identifiers (URIs) together with HTTP get, post, put and delete methods. Unlike REST, CoAP runs over User Datagram Protocol (UDP) instead of TCP which makes it suitable for full IP networking in small micro-controllers. Retries and reordering are implemented at the application stack using a messaging sub-layer that detects duplicated messages and provides reliable communication using different types of messages. Confirmable messages must be acknowledged by the receiver, non-confirmable follow the fire and forget model. While being a lightweight protocol, CoAP still provides important features:

- Resource Observation - CoAP can extend the HTTP request model with the ability to observe a resource therefore monitoring resources of interest using a publish/subscribe mechanism.

- Resource Discovery - CoAP servers provide a list of resources using well-known URIs that allow clients to discover what resources are provided and their types.

- Interoperability - since CoAP is based on the REST architecture, a simple proxy enables CoAP to easily interoperate with HTTP.

A study that compared CoAP and HTTP using mobile networks concluded that there is no situation where CoAP would consume more resources than HTTP [4]

Message Queue Telemetry Transport (MQTT)

MQTT is a publish/subscribe messaging protocol designed for lightweight Machine to Machine (M2M) communications. It employs a client/server model and consists of three components, the publisher, the subscriber and a broker. Subscribers register their interest for a specific topic and then get informed by the broker when a publisher generates data regarding that topic. Every message is a

discrete chunck of data, opaque to the broker. The broker, on is side, checks authorization of the publishers and subscribers. MQTT supports three Application Level Quality of Service (QoS) levels:

- At Most Once (Fire and Forget): A message will not be acknowledged by the receiver or stored and redelivered by the sender.

- At Least Once: It is guaranteed that the message will be delivered to the receiver, but more that one can reach the destination due to message resending. The sender stores the message until it gets an acknowledge from the receiver.

- Exactly Once: A four-way handshake mechanism is used to guarantee that the message will be received exactly once by the counterpart.

MQTT has support for persistent messages stored on the broker, where the most recent message will be sent to a client that subscribes that topic. Clients can register a custom message to be sent to the broker on disconnect enabling other subscribers to know when a device disconnects. MQTT runs on TCP which in some cases causes drawbacks in performance. A performance evaluation of MQTT and CoAP [5] provides comparisons on several protocol facets:

- Influence of Packet Loss on Delay: With low values of packet loss, MQTT experienced lower delays, but as the packet loss increased CoAP performed better. This is due to the greater TCP overheads involved in the retransmissions of messages when compared to UDP.

- Influence of Packet Loss on Data Transfer: CoAP generated less data for each packet loss versus all the MQTT QoS levels.

- Overheads for Message Sizes: When packet loss rate is low, CoAP generates less overhead than MQTT for all message sizes, but as message size grows, the reverse is true. This happens because when the message size is is large, the probability that UDP loses the message is higher than TCP which causes CoAP to retransmit the whole message more often than MQTT.

In order to address the drawbacks on constrained devices, Message Queue Telemetry Transport for Sensor Networks (MQTT-SN) protocol[6] was created. Among the improvements and new features, MQTT-SN runs on UDP, adds broker support for indexing topic names, provides a discovery procedure to help clients without a pre-configured server address and supports devices in sleep state. With this approach, an extra gateway is necessary to convert from MQTT-SN to MQTT so the communications can be understand by the broker.

3.1.4 Session Layer

So far security issues have not been address in any of the previous layers, this is because security is an expensive, optional feature. The application layer protocols rely on underneath layers to achieve secure communications, and network layer protocols assume that if security in necessary then it has already been handled in upper layer protocols. In fact, the session layer is where the security mechanisms are implemented and provides an abstraction layer to application layer protocols. These mechanisms work on top of the transport layer and aim to provide authentication, confidentiality and message integrity.

Transport Layer Security (TLS)

TLS is a well-known security protocol that is used to provide secure transport layer for TCP communications, allowing the upper layer protocols to be left untouched. TLS operation consists of two phases: the handshake and then the data encryption. During the handshake, both parties negotiate which algorithms will be used during the session, authenticate themselves, and prepare the shared secret for the data encryption. Both HTTP and MQTT work over TCP and use TLS as the adopted security protocol.

Datagram Transport Layer Security (DTLS)

DTLS aims to be the equivalent of TLS over UDP transport layer. DTLS works over datagrams that can be lost, duplicated, or received in the wrong order, therefore needing some extra mechanisms(application layer protocols QoS) to cope with that. Although both CoAP and MQTT-SN work over UDP and use DTLS as the adopted security, some authors argue that DTLS is not a suitable option [7] and defend the need of a new integrated security solution. Some of the presented drawbacks are:

- There is no multicast support, which is a key feature in IoT (topology discovery and update for example).
- Handshake phase is prone to exhaustion attacks on the device resources.
- The loss of a message in-flight requires the retransmission of all the messages in-flight.

A final overview of the analysed protocols and security solutions is given in Table 1. And a comparison of the protocol stack is shown in Table 2.

3.2 Attack Analysis, Detection and Prevention

Exploitation of existing solutions in the forms of malicious attacks can be found at all the studied OSI layers. They can go from the well-known Denial of Service (DoS) at the application layer to a physical intruder replacing some node on a sensor field. However, given the characteristics of the devices and networks used in IoT combined with the power consumption focus of this work, a specific kind of attacks performed at the network layer is of special interest and importance:

Table 1. IoT Application Protocols Comparison

Application Protocol	RESTful	Request/ Response	Publish/ Subscribe	Adjustable QoS	Transport	Security
HTTP	✓	✓	✗	✗	TCP	TLS
CoAP	✓	✓	✓	✓	UDP	DTLS
MQTT	✗	✗	✓	✓	TCP	TLS
MQTT-SN	✗	✗	✓	✓	UDP	DTLS

Table 2. Protocol Stack Comparison Overview

Layer	Web	IoT
Application	HTTP	CoAP
Session	TLS	DTLS
Transport	TCP	UDP
Network	IPv6	6LoWPAN
Data-Link/Phy	802.11	802.15.4

Battery Depletion Attacks aka Vampire Attacks

//pequeno resumo do que são este tipo de ataques

este paper mostra ataques de routing sem mencionar protocolos em especifico: [8, 9]

este paper mostra ataques no protocolo de routing rpl e no 6lowpan em especifico: [10]

3.2.1 Stateless Protocols

Carousel Attack

Stretch Attack

3.2.2 Stateful Protocols

Directional Antenna Attack

Malicious Discovery Attack

4 Proposed Solution

5 Work Evaluation

6 Work Planning

7 Conclusion

References

1. Al-Fuqaha, A., Guizani, M., Mohammadi, M., Aledhari, M., Ayyash, M.: Internet of Things: A Survey on Enabling Technologies, Protocols and Applications. IEEE Communications Surveys & Tutorials **PP**(99) (2015) 1–1
2. Kok Seng Ting, Gee Keng Ee, Chee Kyun Ng, N.K.N., Ali, B.M.: The Performance Evaluation of IEEE 802 . 11 against. (October) (2011) 850–855
3. Hui, J., Culler, D.: Extending IP to low-power, wireless personal area networks. IEEE Internet Computing **12**(4) (2008) 37–45
4. Savolainen, T., Javed, N., Silverajan, B.: Measuring Energy Consumption for RESTful Interactions in 3GPP IoT Nodes. (2014) 1–8
5. Ma, X., Valera, A., Tan, H.x., Tan, C.K.y.: Performance Evaluation of MQTT and CoAP via a Common Middleware. (April) (2014) 21–24
6. Ibm: MQTT For Sensor Networks (MQTT-SN) Protocol Specification. (2013) 28
7. Alghamdi, T.a., Lasebae, A., Aiash, M.: Security analysis of the constrained application protocol in the Internet of Things. 2nd International Conference on Future Generation Communication Technologies, FGCT 2013 (2013) 163–168
8. Vasserman, E.Y., Hopper, N.: Vampire attacks: Draining life from wireless ad Hoc sensor networks. IEEE Transactions on Mobile Computing **12**(2) (2013) 318–332
9. Vanitha, K., Dhivya, V.: A Valuable Secure Protocol to Prevent Vampire Attacks In Wireless Ad Hoc Sensor Networks. **3**(3) (2014)
10. Pongle, P., Chavan, G.: A survey: Attacks on RPL and 6LoWPAN in IoT. 2015 International Conference on Pervasive Computing (ICPC) **00**(c) (2015) 1–6